ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

INGENIERO INDUSTRIAL

SMARTCITY: Sistema de gestión de tráfico inteligente.

Autor:
Carlos Seoane Veiga

Directores:
Álvaro Sánchez Miralles
Enrique Díaz Plaza
Dolores Salcedo Gutiérrez

Madrid
Mayo 2012

Proyecto realizado por el alumno:

Carlos Seoane Veiga

Fdo.: ..... Fecha: ...../...../.....

Autorizada la entrega del proyecto cuya información no es de carácter confidencial

LOS DIRECTORES DEL PROYECTO

Álvaro Sánchez Miralles

Fdo.: ..... Fecha: ...../...../.....

Enrique Díaz Plaza

Fdo.: ..... Fecha: ...../...../.....

Dolores Salcedo Gutiérrez

Fdo.: ..... Fecha: ...../...../.....

VºBº del Coordinador de Proyectos

Álvaro Sánchez Miralles

Fdo.: ..... Fecha: ...../...../.....

SMARTCITY: SISTEMA DE GESTIÓN DE TRÁFICO INTELIGENTE

Autor: Seoane Veiga, Carlos.

Directores: Sánchez Miralles, Álvaro; Díaz Plaza Sanz, Enrique; Salcedo Gutiérrez, Dolores.

Entidades Colaboradoras: IIT (ICAI – Universidad Pontificia de Comillas), IBM España.

RESUMEN DEL PROYECTO

I. Introducción

Desde la segunda mitad del siglo pasado se ha venido incrementando de forma rápida y continuada la tasa de atascos en las grandes urbes. Estos atascos representan una fuente de polución y un sumidero de tiempo, dinero y salud que perjudican seriamente la calidad de vida de todos los habitantes de las grandes ciudades.

Esta repercusión en la calidad de vida de los ciudadanos no es aceptable si existe una tecnología que, utilizada correctamente, permita reducir de forma considerable o eliminar el problema existente.

Los gestores de las ciudades han intentado solucionar el problema utilizando distintas estrategias, y si bien en un primer momento la solución propuesta fue la expansión de la vía, rápidamente se demostró que esta no podía ser la única alternativa a la hora de solucionar el problema.

Así, desde los años 70, se ha venido incrementando el número de intersecciones controladas por semáforos, pero, a pesar de haber conseguido instalar en la red una herramienta que nos permite controlar el tráfico, no se ha prestado la suficiente atención al uso de esta, es decir, no se ha dado la suficiente importancia a las estrategias a través de las cuales se controlan estos semáforos.

En este proyecto se ha desarrollado un sistema de gestión de tráfico reactivo al estado de este que funciona de forma totalmente automática. Esto se ha conseguido a través de una sensorización del área en la que se pretende instalar el sistema, y la inclusión de los datos obtenidos de estos sensores en un modelo

matemático del que se obtendrán los estados de los semáforos a través de los cuales se consigue un tráfico más fluido.

Además, se ha preparado el sistema para ser fácilmente aplicable a una nueva ciudad.

Pese a existir estrategias desarrolladas dentro de esta clasificación, no existe ningún proyecto terminado que pueda presumir de haber logrado desarrollar una estrategia de control de tráfico que funcione en tiempo real, de forma eficiente y que sea posible implementar en un escenario lo suficientemente grande y complejo como una ciudad. Muchas de ellas tienen una eficacia probada, pero en un ámbito muy restringido en el que los escenarios en los que se prueban son muy simples o de un tamaño reducido. Es por esto que, con la implantación de este sistema en un escenario correspondiente a un barrio completo de una ciudad (el Barrio de la Misericordia en Málaga), se ha logrado un gran avance en lo que a sistemas de gestión de tráfico se refiere.

II. Arquitectura

Como se puede ver en la Ilustración 1, la arquitectura del proyecto consta de seis módulos diferenciados.

Cuatro de ellos constituyen el sistema de gestión de tráfico y los dos restantes cumplen la función de automatizar la implantación del sistema en nuevos escenarios.

Los cuatro primeros módulos son el simulador, el agente de control, el optimizador y la interfaz de usuario.

El simulador, SUMO (Simulation of Urban Mobility), es el encargado de llevar a cabo la simulación en un escenario creado a tal efecto y bajo unas condiciones de tráfico que el usuario indica a través de la interfaz de usuario, desarrollada en Microsoft Excel mediante el uso de VBA. Durante esta simulación, al estar situados en el escenario sensores destinados a medir el nivel y las características del tráfico, el agente de control, desarrollado en Python, irá recibiendo estos datos que organizará cada cinco minutos en forma de ratios de giro en cada cruce y número de vehículos entrantes en la zona a controlar. Cuando esta información esté organizada, el agente de control será el encargado de

enviarla al optimizador, IBM ILOG CPLEX, en forma de dos archivos *.txt y ordenar la ejecución de la optimización de la que se obtendrá el estado de los semáforos que minimiza las colas totales en el escenario. Esta configuración de semáforos será enviada de nuevo desde el optimizador al agente de control, que será el encargado de fijar el estado de los diferentes semáforos de acuerdo a esta configuración óptima de semáforos.

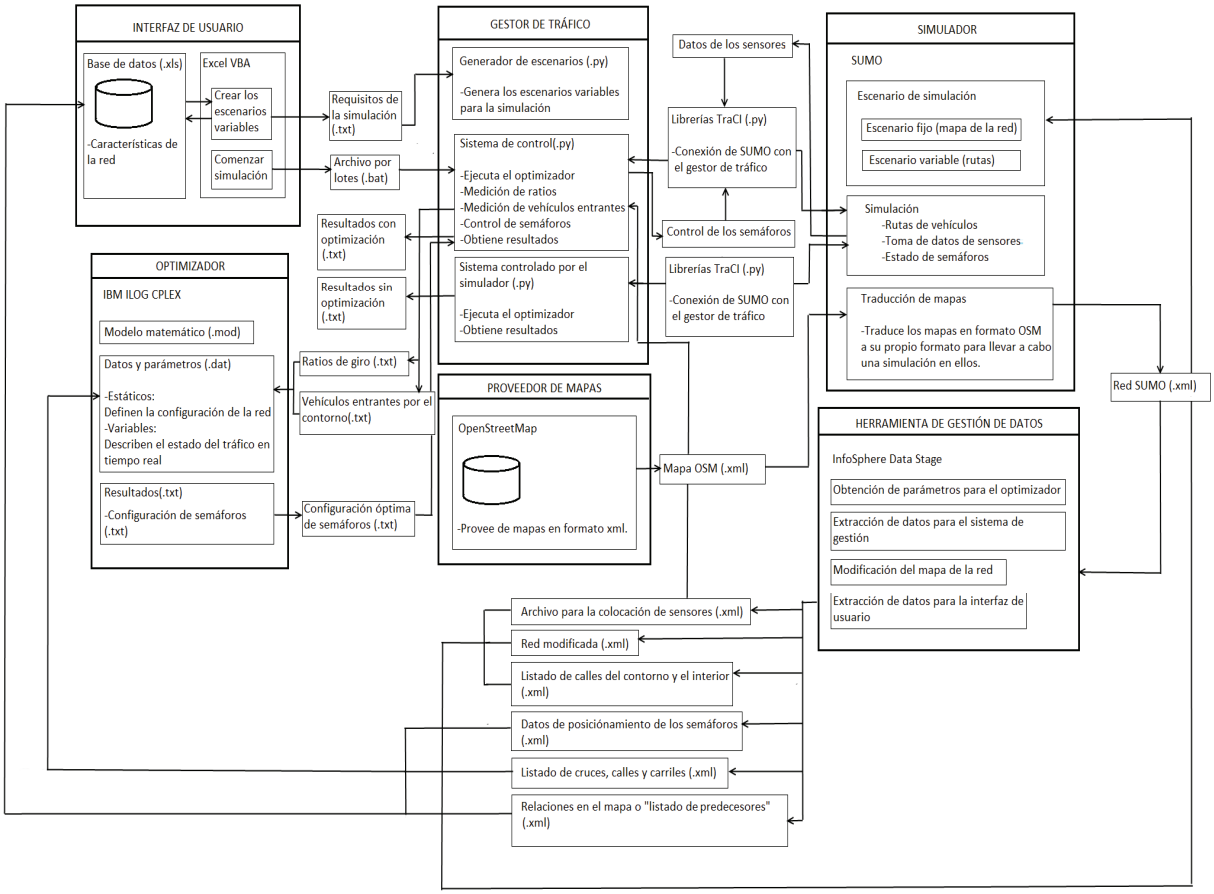

INTERFAZ DE USUARIO
Base de datos (.xls)
-Características de la red
Excel VBA
Crear los escenarios variables
Comenzar simulación
Requisitos de la simulación (.txt)
Archivo por lotes (.bat)
GESTOR DE TRÁFICO
Generador de escenarios (.py)
-Genera los escenarios variables para la simulación
Sistema de control(.py)
-Ejecuta el optimizador
-Medición de ratios
-Medición de vehículos entrantes
-Control de semáforos
-Obtiene resultados
Sistema controlado por el simulador (.py)
-Ejecuta el optimizador
-Obtiene resultados
Resultados con optimización (.txt)
Resultados sin optimización (.txt)
Datos de los sensores
Librerías TraCI (.py)
-Conexión de SUMO con el gestor de tráfico
Control de los semáforos
Librerías TraCI (.py)
-Conexión de SUMO con el gestor de tráfico
SIMULADOR
SUMO
Escenario de simulación
Escenario fijo (mapa de la red)
Escenario variable (rutas)
Simulación
-Rutas de vehículos
-Toma de datos de sensores
-Estado de semáforos
Traducción de mapas
-Traduce los mapas en formato OSM a su propio formato para llevar a cabo una simulación en ellos.
Red SUMO (.xml)
OPTIMIZADOR
IBM ILOG CPLEX
Modelo matemático (.mod)
Datos y parámetros (.dat)
-Estáticos:
Definen la configuración de la red
-Variables:
Describen el estado del tráfico en tiempo real
Resultados(.txt)
-Configuración de semáforos (.txt)
Ratios de giro (.txt)
Vehículos entrantes por el contorno(.txt)
Configuración óptima de semáforos (.txt)
PROVEEDOR DE MAPAS
OpenStreetMap
-Provee de mapas en formato xml.
Mapa OSM (.xml)
HERRAMIENTA DE GESTIÓN DE DATOS
InfoSphere Data Stage
Obtención de parámetros para el optimizador
Extracción de datos para el sistema de gestión
Modificación del mapa de la red
Extracción de datos para la interfaz de usuario
Archivo para la colocación de sensores (.xml)
Red modificada (.xml)
Listado de calles del contorno y el interior (.xml)
Datos de posiciónamiento de los semáforos (.xml)
Listado de cruces, calles y carriles (.xml)
Relaciones en el mapa o "listado de predecesores" (.xml)

Ilustración 1. Arquitectura del Proyecto

Los dos módulos restantes están constituidos por el proveedor de mapas online OpenStreetMap, del cual se extraerá el mapa de la nueva zona a controlar en formato *.xml e IBM InfoSphere Data Stage, que, una vez traducido el mapa de la zona a controlar (proceso que se lleva a cabo automáticamente a través de una aplicación del simulador SUMO) para que se puedan realizar sobre el

simulaciones, será el encargado de extraer de este diversos parámetros que se utilizarán tanto para configurar el optimizador como el sistema de control.

III. Pruebas

Las pruebas se han realizado utilizando dos escenarios, un escenario que representa una parte del Barrio de la Misericordia y otro que representa el Barrio de la Misericordia completo. Ambos escenarios se pueden observar en las Ilustraciones 2 y 3 y con la implantación del sistema desarrollado en estos no queda duda acerca de su funcionamiento eficaz en escenarios grandes y complejos.

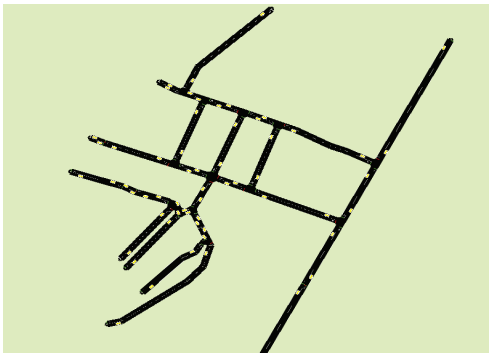


Ilustración 3. Escenario Parcial

Ilustración 2. Escenario correspondiente al Barrio de la Misericordia

IV. *Resultados y conclusiones*

Los resultados obtenidos de la comparación del sistema desarrollado con un sistema de control de lazo abierto (insensible al estado del tráfico) se muestran en la Tabla 1. Para la obtención de resultados se inyectaron vehículos durante media hora de simulación y se empezaron a tomar resultados transcurridos quince minutos, dejando así un espacio de tiempo para la estabilización del tráfico.

La toma de resultados se alargó hasta que el sistema correspondiente fue capaz de despejar la red.

	Sistema de Gestión Inteligente	*Sistema de Gestión de Lazo Abierto*	*Mejoras obtenidas*
Nº de coches en la simulación	352	355	
Tiempo de simulación (s)	1735.00	8173.33	
Tiempo para limpiar la red tras el cese de la creación de rutas (s)	835.00	7273.33	88.52%
Nº medio de vehículos en la red	25.12	47.64	47.26%
Tiempo medio de permanencia de los vehículos en la red (s)	103.84	593.40	82.50%
CO2 emitido (g)	67.70	122.65	44.80%
Combustible consumido (l)	26.87	46.41	42.10%

Tabla 1. Resultados del proyecto

Los resultados mostrados en la Tabla 1 son en realidad la media de los resultados obtenidos para distintos casos de simulación. A lo largo de este proceso de toma de resultados se pudo ir comprobando como el sistema trabaja de forma autónoma, consiguiendo un tráfico más fluido, consiguiendo evitar la formación de atascos que en otros sistemas si tendrían lugar ante iguales flujos vehiculares y reduciendo las emisiones de de CO_2 y el consumo de combustible.

Así, y ante los resultados expuestos, no se puede llegar a otra conclusión que no sea la de que este sistema es tan válido como necesario para las grandes ciudades.

SMARTCITY: SYSTEM FOR SMART TRAFFIC MANAGEMENT

Author: Seoane Veiga, Carlos.

Managers: Sánchez Miralles, Álvaro; Díaz Plaza Sanz, Enrique; Salcedo Gutiérrez, Dolores.

Collaborating Entities: IIT (ICAI – Pontifical University of Comillas), IBM Spain.

ABSTRACT

I. Introduction

Since the second half of the last century, rate of traffic jams has been increasing rapidly and continuously. These traffic jams are a pollution source and a drain of time, money and health, which seriously impact in the quality of life of all citizens of large cities.

This impact in quality of life of citizens is not acceptable if there is a technology that, used properly, allows to significantly reduce or eliminate the existing problem.

Thus, this project stems from the conviction that it is possible with current technology to design a system that provides a more efficient traffic control using traffic lights as a tool to influence this traffic. This will require that traffic lights control strategy takes into account the traffic conditions in real time.

Throughout the project, was developed a traffic control strategy for use on a large, complex stage such as a city and, unlike the strategies currently in use (open-loop strategies), it takes into account the current traffic state to decide on the optimal configuration of states of traffic lights (closed-loop strategy).

Despite the existence of strategies developed within this classification, there is no completed project that can boast of having succeeded in developing a control strategy that operates in real time, adaptive to the traffic state, efficiently and able to implement in a sufficiently large and complex scenario such as a city. Many of these closed loop strategies as MOTION, TUC, OPAC ... had proven efficacy, but in a very restricted area in which the scenarios that are tested are very simple or small sizes. That is why, with the implementation of this system in a scenario corresponding to an entire neighborhood of a city (the district Barrio de la

Misericordia in Málaga), has been achieved a breakthrough as far as traffic management systems are concerned.

During this project was developed a traffic management system adaptive to traffic conditions which works fully automatically. This has been accomplished through installation of sensors in the control area in which it intends to install the system, and including data obtained from these sensors in a mathematical model that will obtain the traffic lights state through which traffic is more fluent.

Furthermore, the system is prepared to be easily applicable to any new city.

With the system developed, have been fully covered the project's objectives, which were established as:

-Optimize traffic management in a scenario that corresponds to a full neighborhood of a city.

-Design an architecture for its implementation sensitive to traffic conditions.

II. Architecture

As shown in Figure 1, project´s architecture consists of six distinct modules. Four of them are the traffic management system and the remaining two have the function to automate the implementation of the system in new nets.

The first four modules are the simulator, the control agent, the optimizer and the user´s interface.

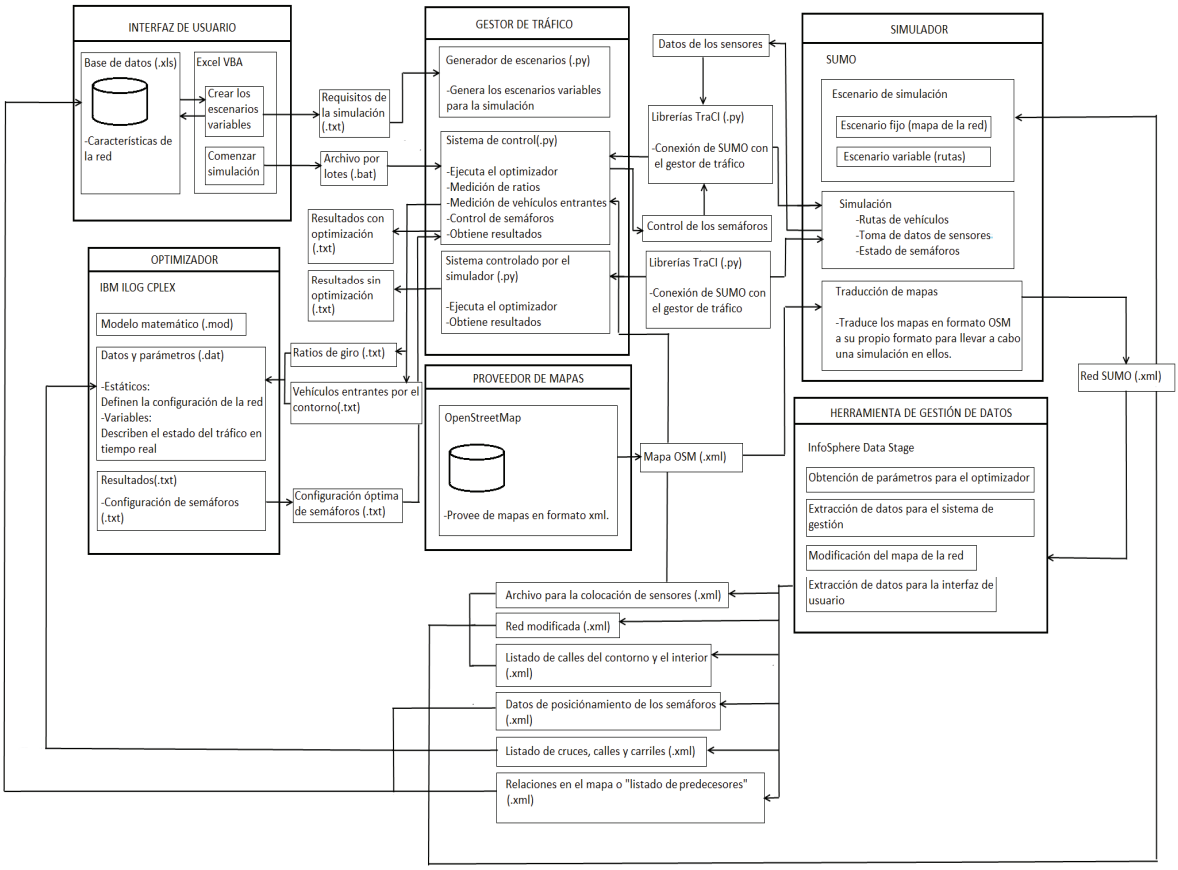

INTERFAZ DE USUARIO
Base de datos (.xls)
-Características de la red
Excel VBA
Crear los escenarios variables
Comenzar simulación
Requisitos de la simulación (.txt)
Archivo por lotes (.bat)
GESTOR DE TRÁFICO
Generador de escenarios (.py)
-Genera los escenarios variables para la simulación
Sistema de control(.py)
-Ejecuta el optimizador
-Medición de ratios
-Medición de vehículos entrantes
-Control de semáforos
-Obtiene resultados
Sistema controlado por el simulador (.py)
-Ejecuta el optimizador
-Obtiene resultados
Resultados con optimización (.txt)
Resultados sin optimización (.txt)
OPTIMIZADOR
IBM ILOG CPLEX
Modelo matemático (.mod)
Datos y parámetros (.dat)
-Estáticos:
Definen la configuración de la red
-Variables:
Describen el estado del tráfico en tiempo real
Resultados(.txt)
-Configuración de semáforos (.txt)
Ratios de giro (.txt)
Vehículos entrantes por el contorno(.txt)
Configuración óptima de semáforos (.txt)
PROVEEDOR DE MAPAS
OpenStreetMap
-Provee de mapas en formato xml.
Datos de los sensores
Librerías TraCI (.py)
-Conexión de SUMO con el gestor de tráfico
Control de los semáforos
Librerías TraCI (.py)
-Conexión de SUMO con el gestor de tráfico
Mapa OSM (.xml)
SIMULADOR
SUMO
Escenario de simulación
Escenario fijo (mapa de la red)
Escenario variable (rutas)
Simulación
-Rutas de vehículos
-Toma de datos de sensores
-Estado de semáforos
Traducción de mapas
-Traduce los mapas en formato OSM a su propio formato para llevar a cabo una simulación en ellos.
Red SUMO (.xml)
HERRAMIENTA DE GESTIÓN DE DATOS
InfoSphere Data Stage
Obtención de parámetros para el optimizador
Extracción de datos para el sistema de gestión
Modificación del mapa de la red
Extracción de datos para la interfaz de usuario
Archivo para la colocación de sensores (.xml)
Red modificada (.xml)
Listado de calles del contorno y el interior (.xml)
Datos de posiciónamiento de los semáforos (.xml)
Listado de cruces, calles y carriles (.xml)
Relaciones en el mapa o "listado de predecesores" (.xml)

Figure 1. Project´s architecture

The simulator, SUMO (Simulation of Urban Mobility), is responsible for carrying out the simulation on a net created for this purpose and under traffic conditions that the user indicates via user´s interface, developed in Microsoft Excel using VBA. During this simulation, sensors located on the stage measure the level and nature of traffic conditions, thus, control agent developed in Python will be receiving this data to organize it every five minutes in the form of turn ratios at each intersection and number of incoming vehicles in the area to be controlled.

When this information is organized, control agent will be responsible for sending it to the optimizer, IBM ILOG CPLEX, as two files *.txt and order the execution of optimization which will obtain traffic lights state that minimize total queue on stage. This set of traffic lights will be sent back from the optimizer to control agent, which will be responsible for establishing the status of the different lights according to the optimal configuration of traffic lights.

Two remaining modules consist of the online map provider OpenStreetMap, from which the map of the new control area is extracted in *. xml format and IBM InfoSphere Data Stage, which, once translated the map (process being carried out automatically through an application of the simulator SUMO) so that simulations can be performed over it, will be responsible for extracting various parameters that will be used to configure both the optimizer and the control system.

III. Tests

Tests were carried out using two scenarios, a scenario that represents a portion of the "Barrio de la Misericordia" and one representing the full neighborhood "Barrio de la Misericordia". Both scenarios can be seen in Figures 2 and 3 and the implementation of the system developed in these leaves no doubt about its effective operation in large and complex scenarios.

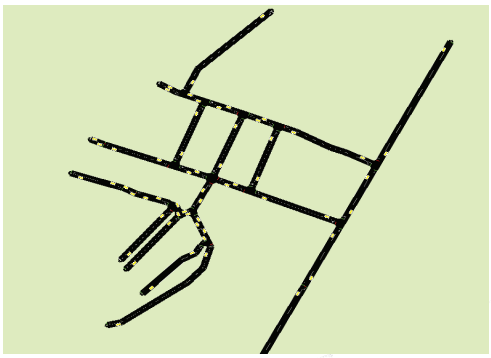


Figure 1. Portion of Barrio de la Misericordia

Figure 2. Barrio de la Misericordia

IV. Results and conclusions

Results obtained from the comparison of the system developed with an open-loop system of control (insensitive to traffic state) are shown in Table 1. To obtain results, vehicles were injected for a half hour and simulation results began to be toke after fifteen minutes, leaving in that way a space of time to stabilize the traffic.

Results taking was lengthened until the corresponding system was able to clear the network.

	Smart Management System	*Open-loop Management System*	*Improvements obtained*
Number of cars during simulation	352	355	
Simulation time (s)	1735.00	8173.33	
Time to clean the network after the finishing the routes injection (s)	835.00	7273.33	88.52%
Average number of vehicles on the network	25.12	47.64	47.26%
Average time spent by vehicles on the network (s)	103.84	593.40	82.50%
Emitted CO2 (g)	67.70	122.65	44.80%
Fuel consumption (l)	26.87	46.41	42.10%

Table 1. Project results

Results shown in Table 1 are actually the average of the results obtained for different simulation cases. Throughout this process of obtaining results was able to check how the system works getting a fluent traffic, avoiding traffic jams which would be formatted in other systems if it would occur the same traffic flow and reducing emissions of CO_2 and fuel consumption.

Thus, and with the results shown, no other conclusion can be obtained than that this system is as valid as necessary for large cities.

ABSTRACT

AGRADECIMIENTOS

A mis padres, por haber estado ahí siempre y hacerme sentir el convencimiento de que siempre será así.

A mi hermana Laura, porque siempre nos apoyaremos el uno al otro.

A mi novia, por hacerme todos los días un poco más fáciles con su sonrisa.

Y por último a mi abuelo Venancio, por haber confiado siempre en mí.

ÍNDICE DE DOCUMENTOS

DOCUMENTO I. MEMORIA

I. Memoria	pág. 15 a 124	110 páginas
II. Manuales de usuario	pág. 125 a 142	18 páginas

DOCUMENTO II. PRESUPUESTO

1. Mediciones	pág. 5	1 página
2. Precios unitarios	pág. 7	1 página
3. Sumas parciales	pág. 9	1 página
4. Sumas parciales	pág. 11	1 página

Documento I

MEMORIA

ÍNDICE

Parte I Memoria *15*

Capítulo 1 Introducción *17*

1 Estado del arte **17**

1.1 Estrategias de tiempo fijo o lazo abierto 19

1.2 Estrategias sensibles al tráfico o de lazo cerrado 20

2 Motivación del proyecto **23**

3 Objetivos **26**

4 Recursos y metodología **26**

4.1 Recursos 26

4.2 Metodología 30

Capítulo 2 Estado inicial del proyecto *33*

1 Introducción **33**

2 Arquitectura **34**

3 Simulador **35**

3.1 Detectores 36

3.2 Semáforos 38

3.3 Aplicaciones ejecutables 39

4 Escenarios **41**

5 Sistema de Control **42**

6 Modelo matemático **42**

7 Generador de escenarios **43**

Capítulo 3 Estado actual del proyecto *45*

1 Introducción **45**

2 Arquitectura **46**

3 Sistema de Control **49**

4 Modelo Matemático **53**

5 Escenarios y extracción de datos **54**

Capítulo 4 Modelo matemático *59*

1 Introducción **59**

2 Índices **60**

3 Parámetros **63**

4 Variables **65**

5 Función objetivo **67**

6 Restricciones **68**

7 Scripts de postprocesamiento **74**

Capítulo 5 Automatización del proceso de implantación del sistema *77*

1 Automatización de la creación de escenarios **77**

2 Parámetros necesarios y método de obtención **79**

2.1 Especificaciones para la extracción de datos 80

2.2 Especificación de los ficheros de salida 89

2.2.1 Listado de las calles y cruces 90

2.2.2 Número total de cruces y calles 91

2.2.3 Listado de predecesores 91

2.2.4 Listado de carriles que van o vienen del contorno 92

2.2.5 Listado de calles en dirección 1 y de calles en dirección 2 92

2.2.6 Listado de cruces sin semaforizar 93

2.2.7 Listado del número de semáforos en cada carril 93

2.2.8 Matriz semáforos lane 94

2.2.9 Vector Semáforos Cruce 94
2.2.10 Listados de estándar y de contorno 95
2.2.11 Listado de vectores 97
2.2.12 Modificación Red para hacer coincidir los linkno 98
2.3 Ejecución del proceso 101

Capítulo 6 Resultados 103

1 Introducción 103

2 Resultados para el Caso 1 106

3 Resultados para el Caso 2 108

4 Resultados para el Caso 3 110

5 Resultados para el Caso 4 112

Capítulo 7 Conclusiones 115

Capítulo 8 Futuros desarrollos 119

1 Simulador 119

2 Capacidad de las vías 120

Bibliografía 121

Parte II Manual de usuario 125

Capítulo 1 Manual de usuario 127

1 Introducción 127

2 Implementación del sistema en un nuevo escenario 127

3 Realización de pruebas y toma de resultados 138

ÍNDICE DE FIGURAS

Ilustración 1. Estrategia de lazo abierto.....19
Ilustración 2. Estrategia de lazo cerrado 20
Ilustración 3. Arquitectura del Proyecto 29
Ilustración 4. Arquitectura Inicial del Proyecto 35
Ilustración 5. Detalle de un sensor de tipo E1 38
Ilustración 6. Detalle de un cruce semaforizado 39
Ilustración 7. Escenario Sencillo 41
Ilustración 8. Arquitectura detallada del proyecto.....47
Ilustración 9. Escenario Sencillo semaforizado 56
Ilustración 10. Escenario Parcial 56
Ilustración 11. Escenario correspondiente al Barrio Completo.....57
Ilustración 12. Detalle de la extracción de un mapa desde OpenStreetMap 78
Ilustración 14. Consumo de combustible frente al tiempo para el Caso1 107
Ilustración 13. Emisiones de CO2 frente al tiempo para el Caso 1.....107
Ilustración 15. Emisiones de CO2 frente al tiempo para el Caso 2.....109
Ilustración 16. Consumo de combustible frente al tiempo para el Caso 2109
Ilustración 17. Origen y destino de la ruta especial simulada para el Caso 3 110
Ilustración 18. Origen y destino de la ruta especial simulada para el Caso 4 113
Ilustración 19. Detalle del proceso de colocación de semáforos en OSM.128

Ilustración 20. Detalle de la utilización de WinSCP.....130

Ilustración 21. Detalle de la utilización de PuTTY 131

Ilustración 22. Detalle de la modificación de la hoja de Excel.....136

Ilustración 23. Detalle de la interfaz de usuario lista para llevar a cabo sobre ella una simulación.....140

Ilustración 24. Detalle de una simulación en el Barrio de la Misericordia142

ÍNDICE DE TABLAS

Tabla 1. Nivel de complejidad de los escenarios.....55

Tabla 2. Probabilidades de generación de "n" rutas en un segundo cualquiera para los tres casos simulados 105

Tabla 3. Resultados para el Caso 1 106

Tabla 4. Resultados para el Caso 2 108

Tabla 5. Resultados para el Caso 3 111

Tabla 6. Resultados para el Caso 4 114

ÍNDICE DE ECUACIONES

Ecuación 1. Factor de corrección k dependiente de la velocidad. 52
Ecuación 2. Índice "i" 60
Ecuación 3. Índice "j" 60
Ecuación 4. Índice "t" 60
Ecuación 5. Índice "c" 61
Ecuación 6. Subespacio "$LContorno$" 61
Ecuación 7. Subespacio "$LInterior$" 61
Ecuación 8. Subespacio "$LDir1$" 62
Ecuación 9. Subespacio "$LDir2$" 62
Ecuación 10. Subespacio "$LPredecesoresc$" 62
Ecuación 11. Subespacio "$LCarrilesConPredec$" 63
Ecuación 12. Subespacio "$LInteriorPredecesores$" 63
Ecuación 13. Subespacio "$LInteriorSinPredecesores$" 63
Ecuación 14. Parámetro "*NbSemaforos(c)*" 64
Ecuación 15. Parámetro "$NbSemaforosCrucei$" 64
Ecuación 16. Parámetro "$EntrantesContornoc$" 64
Ecuación 17. Parámetro "FV" 64
Ecuación 18. Parámetro "*Tperiodo*" 65
Ecuación 19. Parámetro "$TminV$" 65
Ecuación 20. Parámetro "*TSemSteady*" 65
Ecuación 21. Parámetro "*Tdesp*" 65
Ecuación 22. Parámetro "*NumGrande*" 65

Ecuación 23. Parámetro *"Ratios(c)"* 65

Ecuación 24. Variable *"Sem(t,c)"* 66

Ecuación 25. Variable *"Colas(t,c)"* 66

Ecuación 26. Variable *"FlujoSalMax(t,c)"* 66

Ecuación 27. Variable *"FlujoSal(t,c)"* 66

Ecuación 28. Variable *"FlujoEnt(t,c)"* 67

Ecuación 29. Variable *"AlgunV1(t,i)"* 67

Ecuación 30. Variable *"AlgunV2(t,i)"* 67

Ecuación 31. Función Objetivo 67

Ecuación 32. Definición de las colas (1) 68

Ecuación 33. Definición de las colas (2) 68

Ecuación 34. Definición de las colas (3) 68

Ecuación 35. Asignación de *"Sem(0,c)"* 69

Ecuación 36. Valor de *"FlujoSalMax(t,c)"* para carriles controlados por semáforos....70

Ecuación 37. Valor de *"FlujoSalMax(t,c)"* para carriles no controlados por semáforos....70

Ecuación 38. Valor de *"FlujoEnt(t,c)"* si "c" pertenece al contorno....70

Ecuación 39. Valor de *"FlujoEnt(t,c)"* si "c" pertenece al interior 71

Ecuación 40. Valor de *"FlujoEnt(t,c)"* si "c" es del interior y "t" es menor que *"Tdesp"* 71

Ecuación 41. Valor de *"FlujoEnt(t,c)"* si "c" es del interior y no tiene predecesores....71

Ecuación 42. Establecimiento del valor de *"Sem(t,c)"* para los semáforos en dirección 1. 72

Ecuación 43. Establecimiento del valor de *"Sem(t,c)"* para los semáforos en dirección 2. 72

Ecuación 44. Restricción de seguridad para los cruces 72

Ecuación 45. Restricción de tiempo mínimo en verde..... 72

Ecuación 46. Restricción para la continuidad del semáforo en verde..... 73

Ecuación 47. Identificación y fijación de estado de los semáforos de los cruces que solo tienen semáforos solo en la dirección 1 74

Ecuación 48. Identificación y fijación de estado de los semáforos de los cruces que solo tienen semáforos solo en la dirección 2 74

Ecuación 49. Probabilidad de creación de rutas en función del parámetro k 104

Parte I MEMORIA

Capítulo 1 INTRODUCCIÓN

En este capítulo se realiza una introducción con el objetivo de contextualizar el proyecto realizado. Se comienza con el estado del arte, apartado en el que se trata el estado actual de los sistemas de control de tráfico. En el apartado de motivación se explica el interés en el desarrollo de este proyecto, es decir, a que necesidades responde la solución tecnológica presentada. Por último, se tratan los objetivos establecidos y la metodología y los recursos necesarios para alcanzarlos.

1 Estado del arte

En este apartado del proyecto se atenderá a las principales estrategias de control del tráfico desarrolladas hasta el momento. Estas estrategias se pueden clasificar de varias formas [1]-[6].

La primera distinción a realizar entre estas estrategias atiende al tipo de datos en los que se basan para levar a cabo la optimización. Hay dos tipos de estrategias atendiendo a este criterio:

Estrategias de tiempo fijo o lazo abierto: Son estrategias desarrolladas a partir de datos históricos de flujo y densidad vehicular (esto no impide hacer una distinción entre distintos periodos de tiempo a lo largo del día). La consideración de que esta estrategia es capaz de gestionar de forma más o menos óptima el tráfico implica asumir que estos datos tendrán una variación muy leve para tiempos largos. Los sistemas de control de este tipo no reciben ninguna información del estado

del tráfico actual. Se caracterizan por un bajo coste de implantación (no existe la necesidad de colocación de sensores,…), una baja eficiencia en zonas de alta variabilidad de la demanda, una capacidad nula de respuesta ante accidentes o eventos, y su aplicabilidad limitada a condiciones en las que la red no está saturada.

Estrategias sensibles al tráfico o de lazo cerrado: Son estrategias en las que las herramientas de control del tráfico (semáforos principalmente) se utilizan de acuerdo a mediciones en tiempo real del estado del tráfico. Esta información es enviada al sistema de control mediante sensores que pueden ser de bucle inductivo, de análisis de señales de video, etc., lo que incrementa su coste de implantación. Pese a ello, estas estrategias se encuentran particularmente interesantes ya que disminuirán a largo plazo costes tanto económicos como medioambientales al reducir la tasa de atascos, el tiempo de permanencia de los vehículos en las vías y la peligrosidad en las carreteras.

Una segunda distinción divide los sistemas de control en dos grupos en función de su aplicabilidad:

Sistemas para intersecciones aisladas: Se incluyen en este grupo todos aquellos sistemas de control que solo son aplicables a una o varias intersecciones aisladas. Este tipo de sistemas carece de interés en el ámbito de este proyecto por las limitaciones que presenta para regular el tráfico en las grandes urbes, donde un sistema de control más eficiente es particularmente interesante.

Sistemas para la coordinación de intersecciones: Se incluyen en este grupo todos aquellos sistemas capaces de gestionar eficientemente una red con varias intersecciones. Dentro de este grupo se puede hacer una segunda distinción atendiendo a si el sistema de control funciona de

forma centralizada o descentralizada, es decir, atendiendo a si la "inteligencia" del sistema se concentra en un único punto, o, por el contrario, está distribuida a lo largo de toda la red.

Siguiendo las clasificaciones propuestas se pueden revisar las principales estrategias de control de tráfico desarrolladas hasta el momento:

1.1 Estrategias de tiempo fijo o lazo abierto

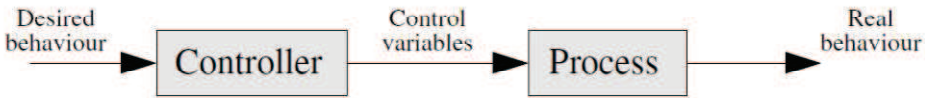


Ilustración 1. Estrategia de lazo abierto

Son las estrategias más utilizadas por sus bajos costes de instalación y mantenimiento y su buen desempeño bajo condiciones de red no saturada.

Dentro de las estrategias de lazo abierto para **intersecciones aisladas** destacan los sistemas **SIGSET** [7] y **SIGCAP** [8]. La distinción entre ambos radica en que SIGSET trata de minimizar el tiempo de espera total mientras que SIGCAP maximiza la capacidad de las intersecciones.

En el caso de sistemas **para la coordinación de intersecciones** destacan tres sistemas de control:

MAXBAND [9] y **UTCS** [10]: Ambos maximizan el ancho de banda de las avenidas principales, es decir, su capacidad para dar servicio a un

número determinado de vehículos. Cabe destacar que MAXBAND introduce el concepto de "olas de verde" en un intento por maximizar el número de vehículos que pueden atravesar una vía principal a una velocidad determinada sin realizar ninguna parada.

TRANSYT [11]: Lleva a cabo una optimización por gradiente, optimiza siguiendo un sistema iterativo.

1.2 Estrategias sensibles al tráfico o de lazo cerrado

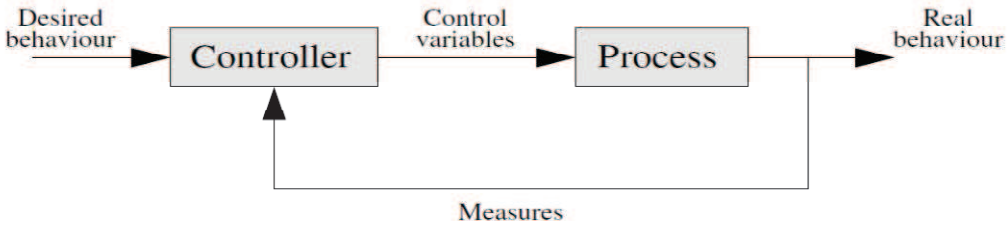


Ilustración 2. Estrategia de lazo cerrado

Dentro de las estrategias sensibles al tráfico para **intersecciones aisladas** [12]-[16] destacan el "**Método de Intervalo de Vehículos**" y su versión mejorada **MOVA** (Microproccesor Optimised Vehicle Actuation) [17]. Ambos se basan en procesos iterativos, pero mientras el "Método de Intervalo de Vehículos" se basa en incrementar el tiempo de verde si se detecta un vehículo dentro del "intervalo crítico de tiempo", MOVA realiza un análisis de costes y beneficios para decidir si el cambio de fase se debe dar en esta o la siguiente iteración, permitiendo así dar prioridad a un tipo de vehículo frente a otros, particularidad que puede ser de especial interés en el caso de los buses urbanos o vehículos pertenecientes

a servicios públicos en general. De esta forma, MOVA actuará en estado de saturación de la vía, es decir, cuando no es posible vaciar una cola al finalizar un tiempo de verde, maximizando su capacidad y en estado de no saturación minimizando los retrasos y paradas.

Las estrategias sensibles al tráfico y **aplicables a la coordinación de intersecciones** representan las estrategias de mayor interés para este proyecto y se caracterizan por unos costes de implantación, operación, mantenimiento y expansión más elevados debido a la necesidad de utilización de sensores que nos proporcionen la información necesaria para resolver la optimización en tiempo real. Los mayores problemas encontrados en este campo son la elevada capacidad de computación que los sistemas desarrollados hasta el momento requieren, así como la disminución de la eficiencia observada en estos sistemas cuando la red se encuentra en estado de saturación. Pese a existir estrategias desarrolladas dentro de esta clasificación, no existe ningún proyecto terminado que pueda presumir de haber logrado desarrollar una estrategia de control de tráfico que funcione en tiempo real, de forma eficiente y que sea posible implementar en un escenario lo suficientemente grande y complejo como una ciudad. Muchas de ellas tienen una eficacia probada, pero en un ámbito muy restringido, en el que los escenarios en los que se prueban son muy simples o de un tamaño reducido. Dentro de esta clasificación se puede atender a:

Las **estrategias centralizadas** entre las que destacan:

SCOOT (Split, Cycle and Offset Optimization Technique) [18]-[19]: Es considerada la versión sensible al tráfico de TRANSYT por llevar a cabo también un proceso iterativo. Esta estrategia trata de minimizar un PI (Performance Index) que representa de forma conjunta los retrasos, la

longitud de las colas, y las paradas realizadas por los vehículos. Lleva a cabo tres optimizaciones, una para el tiempo de ciclo, otra para la "Fracción de Tiempo en Verde" o "Green Split" y una tercera para los "Offsets" o tiempo entre los verdes de semáforos contiguos. Actualmente ofrece la posibilidad de otorgar prioridad a autobuses.

SCATS (Sydney Cooridinated Adaptive Traffic System) [20]: Esta estrategia busca un plan de coordinación de semáforos en una librería interna en función de las condiciones de la red. La inclusión de vehículos prioritarios solo ha sido probada en simulación.

RHODES (Real-time Hierarchical Optimizated Distributed and Effective System) [21]: Este sistema distingue tres niveles de jerarquía y se caracteriza por responder al comportamiento estocástico del tráfico. Puede otorgar prioridad a autobuses a través de BUSBAND.

MOTION [22]: Trabaja creando primero una red de olas de verde de forma centralizada. En una segunda optimización ajusta los tiempos dejando un pequeño tiempo de repuesto para realizar un último ajuste de forma local en cada cruce.

TUC: Este sistema basa su estrategia en mantener las colas dentro de unos límites para evitar que la red sature.

Las **estrategias descentralizadas** como:

UTOPIA: Funciona a través de un control central que predice el estado del tráfico en toda la red cada 15 minutos. Con esta información calcula los tiempos óptimos y los envía a los controladores locales, que se ocupan de una pequeña parte de la red. Estos controladores locales se comunican con otros controladores locales cercanos, obteniendo así cada uno de ellos un nuevo óptimo local. Cualquier desviación que los

controladores locales introduzcan respecto a lo indicado por el controlador central será penalizada.

OPAC [24]: Trabaja dividendo la red a controlar en subredes, y funciona a través de dos módulos, uno para períodos de congestión y otro que entra en funcionamiento cuando la red no está saturada.

2 Motivación del proyecto

Desde la segunda mitad del siglo pasado se ha venido incrementando de forma rápida y continuada la tasa de atascos en la mayoría de las grandes urbes.

Estos atascos representan una fuente de polución y un sumidero de tiempo, dinero y salud que perjudican de forma directa la calidad de vida de todos los habitantes de las grandes ciudades.

Esta repercusión que se da en la calidad de vida de los ciudadanos no puede permitirse si existe una forma eficiente de solucionar el problema.

La primera solución que los gestores de las ciudades propusieron para solucionar el problema fue la expansión de la vía, pero rápidamente se demostró que esta no podía ser la única alternativa para solucionarlo.

Desde los años 70 se ha optado por una nueva solución consistente en incrementar el número de intersecciones controladas por semáforos pero no se ha puesto el suficiente interés en mejorar la eficacia de las estrategias de control de estos.

Además, las predicciones que indican que en el año 2050 el setenta y cinco por ciento de la población vivirá concentrada en grandes urbes

junto con una reflexión acerca de no solo el sistema de gestión de tráfico, sino del modelo de consumo y funcionamiento de las ciudades actuales, nos lleva a la conclusión de que este modelo debe cambiar, ya que no será capaz de soportar este incremento de concentración de la población a la par que esta aumenta sin que se vea repercutida enormemente la calidad de vida de sus ciudadanos.

De esta reflexión nace la idea de empezar a construir las "Smart Cities" basándose en un modelo de ciudad que permita llegar a los niveles de concentración de la población estimados para el futuro cumpliendo las siguientes premisas:

-No dañar el medio ambiente

-Llevar a cabo una gestión inteligente de las ciudades y los servicios que estas ofertan basada en las tecnologías de la información y comunicación existentes.

-Realizar esta transformación de las ciudades teniendo como fin último el desarrollo sostenible.

Dentro de este marco de "Smart Cities" uno de los aspectos a modificar para construir una ciudad sostenible es el modelo de gestión del el tráfico, que, como se ha comentado anteriormente, ya ha empezado a mostrar sus carencias en los últimos años.

Es por eso que este proyecto nace con la intención de ofrecer un sistema de control de tráfico de lazo cerrado que pueda trabajar coordinando cruces en un escenario complejo.

La decisión de la realización de este parte del estado precario de las estrategias de control del tráfico en las ciudades y la baja eficiencia de las

soluciones tecnológicas disponibles hasta el momento, especialmente bajo condiciones de saturación de la red.

Con este proyecto se pretende:

-Ayudar a que el transporte sea más rápido y económico, suponiendo un ahorro importante en tiempo y dinero para cada conductor.

-Reducir los niveles de polución en las ciudades de forma notable, ayudando a mitigar un problema por desgracia cada vez más frecuente que se refleja, por ejemplo, en el cierre al tráfico de ciudades como Milán en el año 2011.

-Reducir los niveles de ira en la carretera y estrés en los conductores provocados por la elevada tasa de atascos, mejorando la salud de estos y reduciendo la peligrosidad en la carretera al hacer del transporte interurbano algo más rápido y eficaz.

Anteriormente, Alejandro Saiz Peguero llevó a cabo un proyecto de características similares, pero la complejidad de este, así como la necesidad de estudiar lenguajes de programación como Python, VBA, IBM ILOG CPLEX,… hicieron que no fuese posible completarlo en un solo año, dejando el proyecto con una gran base de programación y arquitectura y una sabia elección de los recursos a emplear.

Por otro lado, se dejaron objetivos básicos sin alcanzar como el funcionamiento del sistema de control en tiempo real y el funcionamiento del sistema en un escenario complejo (solo llegó a funcionar en un escenario de 6 cruces y bajo la indicación previa de los ratios de giro en cada cruce), haciendo de lo realizado hasta el momento un sistema

incapaz de reaccionar a variaciones en el tráfico en un escenario sencillo, e incapaz de funcionar bajo cualquier premisa en un escenario real.

Este proyecto nace de la intención de ofrecer un sistema que sea capaz de funcionar en tiempo real y en un escenario complejo, y para ello se utilizarán como base algunos de los desarrollos llevados a cabo en el proyecto antes mencionado.

3 Objetivos

Los objetivos de este proyecto son:

1) Optimizar la gestión del tráfico en un escenario que corresponde con un barrio completo de una ciudad.

2) Diseñar una arquitectura que permita su implementación sensible al estado del tráfico.

3) Discriminar en la optimización escenarios de baja densidad de tráfico de los de alta densidad. (Opcional).

4) Integrar dicho sistema en sistemas globales de gestión de "Smart Cities". (Opcional)

4 Recursos y metodología

4.1 Recursos

Para la realización del proyecto se requiere de tres estructuras básicas que se comunicarán entre sí. Estas tres estructuras son el

simulador, la interfaz de usuario y el optimizador, que dotará al sistema de inteligencia.

Además, para automatizar la implantación del sistema en distintos escenarios, utilizaremos una base de datos de mapas online, y una herramienta de transformación.

El **Simulador** es una herramienta básica en el diseño de estrategias de control de tráfico ya que permite observar el comportamiento del tráfico ante éstas, algo que sería imposible sin simuladores debido a la complejidad que caracteriza al tráfico. En este caso se ha elegido utilizar la plataforma **SUMO** (Simulation of Urban MObility). Se ha elegido por ser un simulador microscópico, que describe de una forma más fiel que otro tipo de simuladores (macroscópicos o mesoscópicos) el estado del tráfico, además de disponer de una interfaz gráfica (Graphical User Interface - GUI) que permite observar el comportamiento del tráfico ante nuestra estrategia de control de semáforos, la posición de los sensores y el estado de los semáforos. Este simulador reúne todas las características que debe tener un simulador para el diseño de estrategias de control de tráfico, es decir, cumple con la funcionalidad requerida y la posibilidad de aplicar sobre él de sistemas de gestión inteligente. Además resulta ser de código abierto. Se basa en ficheros de texto *.xml. La red se define en un documento de texto y el simulador la une a un estado de demanda de tráfico definido en otro fichero. Tiene la capacidad de generar ficheros con numerosos datos de la simulación.

La **Interfaz** que relaciona al usuario con el simulador consiste en una herramienta que le pide al usuario que indique en qué condiciones quiere que se realice la simulación, es decir, entre qué horas quiere que se lleve a cabo, con que densidad de tráfico y si quiere que se simule algún

tipo de evento multitudinario, accidente,… que se simulará a través de una determinada cantidad de vehículos recorriendo una ruta prefijada por el usuario. Esta interfaz será desarrollada en **Microsoft Office Excel junto con Visual Basic (VBA)** de forma que se consiga una interfaz sencilla y fácil de utilizar.

La **Herramienta de Optimización** que obtendrá la estrategia de control de semáforos óptima será **IBM ILOG CPLEX Optimization Studio**. El problema se modelará conformando un modelo de programación lineal entero mixto o MIP (Mixed Integer Programming) para reducir al máximo los tiempos de computación teniendo en cuenta las características del tráfico, que obligan a la utilización de variables binarias. El gestor de tráfico le suministrará los datos que representan el estado del tráfico en tiempo real y el optimizador le devolverá el estado óptimo de los semáforos a través de un fichero de texto.

La elección de la plataforma SUMO condiciona la elección de la forma en que se comunicará con el optimizador, ya que SUMO incorpora una librería que permite modificar el estado de los semáforos en tiempo real y obtener de SUMO parámetros recopilados por los sensores (número de vehículos, velocidad de éstos…). Esta librería se denomina TraCI (Traffic Control Interface) y está desarrollada en **Python**. Así, Python se encargará tanto de ejecutar las optimizaciones en el momento adecuado como de intercambiar la información proveniente de ambos.

Por otro lado, para la automatización del proceso de creación de escenarios y la obtención de los parámetros para el modelado que estos escenarios determinan, y que tratan de las conexiones entre cruces, las características de las calles y la colocación de los semáforos, se han utilizado dos herramientas.

En primer lugar, la extracción del mapa se realiza a través de la base de datos online **OpenStreetMap**, proveedor de mapas de software libre desde el que se puede extraer un mapa en un fichero de texto *.xml.

Una vez obtenido este fichero, se transformará en un mapa en el que SUMO pueda llevar a cabo una simulación a través de una herramienta del propio simulador, netconvert.exe. Solo esta transformación no es suficiente, ya que se necesitan realizar en él ciertas transformaciones adicionales y extraer datos que permitan llevar a cabo la optimización en IBM ILOG. Todas estas transformaciones se llevarán a cabo con la herramienta de **IBM InfoSphere DataStage**, una plataforma de información unificada pensada para gestionar y analizar datos y contenido independientemente de su tamaño, de forma rápida, para ofrecerlos en contexto a los usuarios, aplicaciones y procesos.

Las relaciones entre las distintas estructuras aparecen detalladas en la Figura 3.

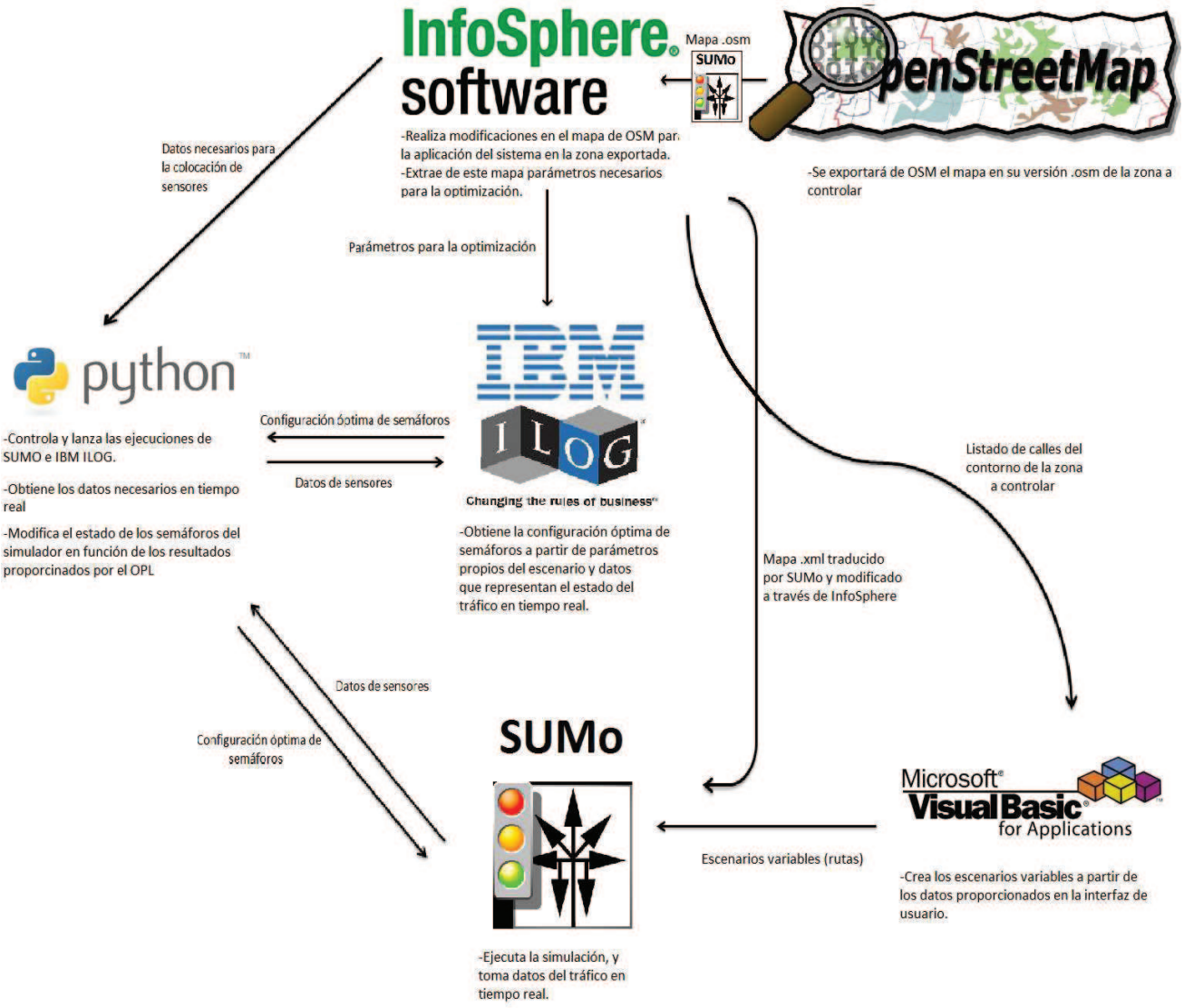

InfoSphere software
Mapa .osm
SUMo
OpenStreetMap
-Realiza modificaciones en el mapa de OSM par. la aplicación del sistema en la zona exportada.
-Extrae de este mapa parámetros necesarios para la optimización.
-Se exportará de OSM el mapa en su versión .osm de la zona a controlar
Datos necesarios para la colocación de sensores
Parámetros para la optimización
python
IBM ILOG
Changing the rules of business
Configuración óptima de semáforos
Datos de sensores
-Controla y lanza las ejecuciones de SUMO e IBM ILOG.
-Obtiene los datos necesarios en tiempo real
-Modifica el estado de los semáforos del simulador en función de los resultados proporcinados por el OPL
-Obtiene la configuración óptima de semáforos a partir de parámetros propios del escenario y datos que representan el estado del tráfico en tiempo real.
Listado de calles del contorno de la zona a controlar
Mapa .xml traducido por SUMo y modificado a través de InfoSphere
Datos de sensores
Configuración óptima de semáforos
SUMo
Microsoft Visual Basic for Applications
Escenarios variables (rutas)
-Crea los escenarios variables a partir de los datos proporcionados en la interfaz de usuario.
-Ejecuta la simulación, y toma datos del tráfico en tiempo real.

Ilustración 3. Arquitectura del Proyecto

4.2 Metodología

Las distintas tareas necesarias para la realización del proyecto se detallan a continuación en orden cronológico:

-Lectura y comprensión de funcionamiento del proyecto realizado el año pasado.

-Documentación del estado actual de la tecnología relacionada con el proyecto y desarrollo del "Estado del Arte".

-Aprendizaje del lenguaje de programación Microsoft VBA.

-Aprendizaje del lenguaje de programación Python.

-Aprendizaje del uso del simulador SUMO.

-Aprendizaje del lenguaje de programación OPL e IBM ILOG Script.

-Modificaciones del escenario, el sistema de control y el modelo matemático para el funcionamiento de la optimización en tiempo real en el Escenario Sencillo.

-Pruebas y solución de problemas para el Escenario Sencillo.

-Aprendizaje del proceso de extracción y transformación de mapas del proveedor de mapas OpenStreetMap.

-Estudio de las transformaciones y obtenciones de datos necesarias para su posterior obtención a través de IBM InfoSphere DataStage.

-Desarrollo de un sistema de control y un nuevo modelo matemático de cara a su aplicación en escenarios más grandes y complejos que los utilizados hasta el momento.

-Puesta en funcionamiento del sistema de gestión inteligente en una parte del Barrio de la Misericordia, Málaga. (Escenario Parcial)

-Pruebas y solución de problemas para el Escenario Parcial.

-Implementación del sistema de control, modelo y generador de escenarios desarrollados en el Barrio de la Misericordia de Málaga.

-Pruebas y solución de problemas para el escenario correspondiente al Barrio de la Misericordia.

-Obtención de resultados.

-Documentación del proyecto.

Capítulo 2 ESTADO INICIAL DEL PROYECTO

Este capítulo tiene como propósito explicar cuál era el estado del que se partía cuando se inició este proyecto. En la primera sección, se explicará cuales eran el alcance y la funcionalidad del proyecto de forma genérica al inicio de este curso. En la segunda sección se explicará cual era la arquitectura previa a los desarrollos llevados a cabo en este proyecto, y del tercer apartado en adelante se dedica una sección a cada uno de los módulos clave del proyecto, comenzando por el simulador, y detallando a continuación los escenarios realizados, el sistema de control, el modelo matemático y el generador de escenarios respectivamente.

1 Introducción

Como se expuso en el capítulo anterior, un proyecto con unos objetivos similares a este fue desarrollado anteriormente por Alejandro Saiz Peguero pero debido a la dificultad que representa la consecución de los objetivos establecidos y a la necesidad de aprendizaje de diferentes lenguajes de programación los desarrollos realizados solamente llegaron a conseguir un sistema de control de tráfico capaz de funcionar en un escenario sencillo compuesto por seis cruces, todos ellos semaforizados. Además, este sistema no era capaz de funcionar en tiempo real de forma reactiva ante cambios en las rutas seguidas por los vehículos, por el contrario, se realizó una observación previa que ayudó a determinar el comportamiento de los vehículos en el escenario, comportamiento que quedó definido a partir de los ratios de giro. La optimización que generaba el estado óptimo de los semáforos fue realizada en base a estos

ratios y a una medida más, realizada en tiempo real, que indicaba el número de vehículos coexistentes en la vía.

2 Arquitectura

Cabe destacar del proyecto anterior el acierto en la elección de la arquitectura, en la que se basa parte de la arquitectura de este año.

La arquitectura elegida consta de cuatro partes diferenciadas sobre las que se trabaja. La primera de ellas, considerada como la de mayor prioridad de decisión fue el simulador (SUMO), encargado de simular y representar la situación del tráfico a nivel microscópico. Esta elección condicionó la del lenguaje en el que se desarrollaría el sistema de control de tráfico (Python), encargado de la gestión de los diferentes módulos y la comunicación entre ellos. Para llevar a cabo la resolución del modelo en función de una serie de parámetros fijos y uno (el número de vehículos en la red) variable, se utilizó el optimizador IBM ILOG CPLEX Optimization Studio. Por último, el generador de rutas o interfaz de usuario fue programado en VBA, consiguiendo de esta forma una interfaz visual y sencilla que facilita la comprobación y utilización del sistema.

La arquitectura que sustentaba al proyecto previo aparece detallada en la Figura 4.

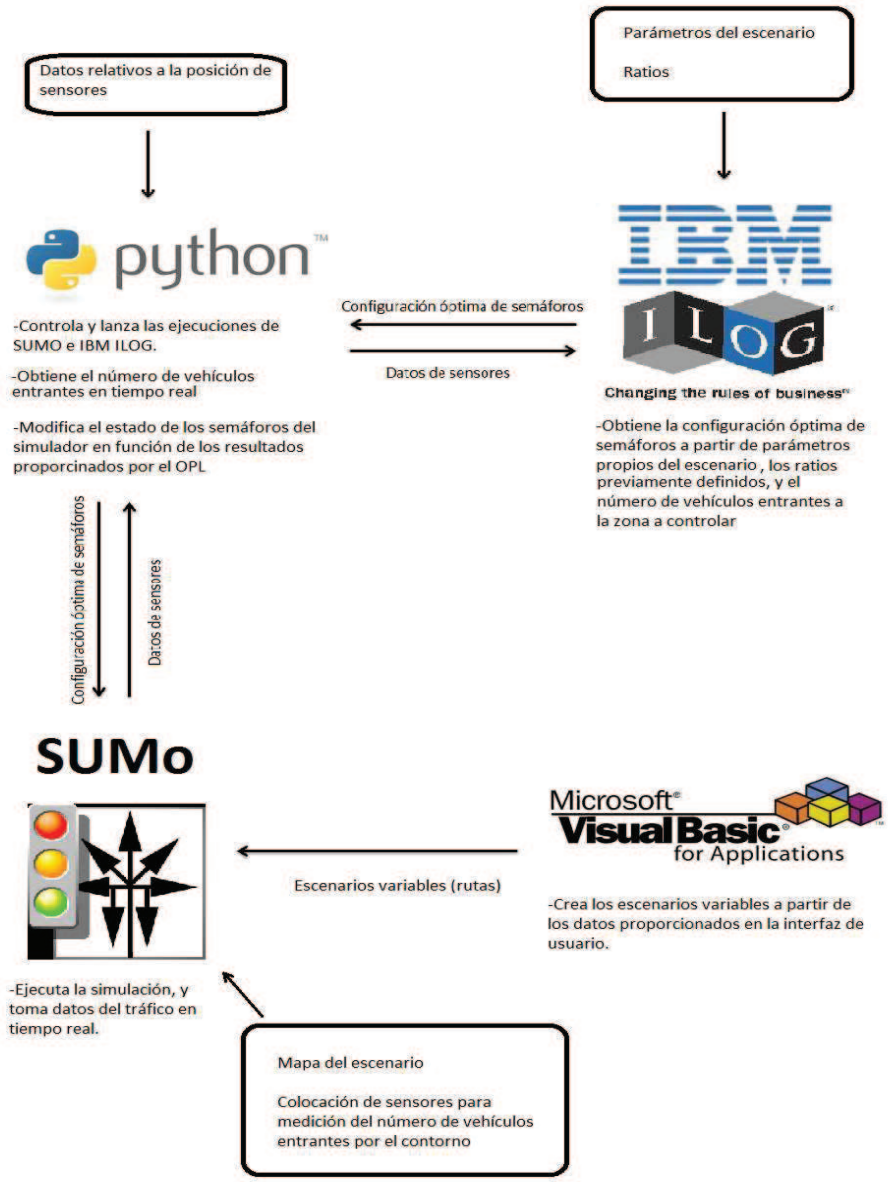

Datos relativos a la posición de sensores
Parámetros del escenario
Ratios
python
-Controla y lanza las ejecuciones de SUMO e IBM ILOG.
-Obtiene el número de vehículos entrantes en tiempo real
-Modifica el estado de los semáforos del simulador en función de los resultados proporcinados por el OPL
Configuración óptima de semáforos
Datos de sensores
IBM ILOG
Changing the rules of business
-Obtiene la configuración óptima de semáforos a partir de parámetros propios del escenario , los ratios previamente definidos, y el número de vehículos entrantes a la zona a controlar
Configuración óptima de semáforos
Datos de sensores
SUMo
Escenarios variables (rutas)
Microsoft Visual Basic for Applications
-Crea los escenarios variables a partir de los datos proporcionados en la interfaz de usuario.
-Ejecuta la simulación, y toma datos del tráfico en tiempo real.
Mapa del escenario
Colocación de sensores para medición del número de vehículos entrantes por el contorno

Ilustración 4. Arquitectura Inicial del Proyecto

3 Simulador

La elección de SUMO como simulador de tráfico y sus motivos se han detallado previamente en el Capítulo 1.

En este capítulo, se intentará profundizar en el funcionamiento de SUMO, dando a conocer sus características y aplicaciones ejecutables útiles que se han utilizado durante el desarrollo del proyecto.

SUMO es un simulador de tráfico multimodal, microscópico y de software libre que permite controlar un amplio número de parámetros relacionados con el tráfico. Gracias a este simulador, podemos cambiar el estado de los semáforos de acuerdo a la estrategia de gestión propuesta o tomar datos del tráfico para llevar a cabo la optimización.

Otro aspecto determinante para poder llevar a cabo el proyecto radica en la posibilidad de que el simulador y el sistema de control (siempre que esté programado en Python) puedan interactuar a través TraCI (Traffic Control Interface) durante la ejecución de la simulación, pudiendo así realizar durante esta la lectura de sensores y controlar los semáforos en tiempo real.

Las características de SUMO que se encontraron más relevantes para la realización del proyecto fueron la posibilidad de colocación de detectores, el hecho de que exista la posibilidad de modificar el estado de los semáforos y la existencia de diferentes aplicaciones ejecutables, cada una encargada de realizar un tipo de tareas.

En estos tres conceptos se profundizó durante el proyecto anterior, dejando así una sólida base para este proyecto en cuanto a la comprensión y utilización del simulador.

3.1 Detectores

SUMO ofrece a sus usuarios la posibilidad de obtener datos de la simulación a través de tres tipos de detectores, cada uno con una funcionalidad distinta.

Los tres tipos de sensores se encuentran en distintos estados de desarrollo, siendo el que se encuentra en una etapa de desarrollo más avanzada el Sensor de Bucle Inductivo (Induction Loop) o E1 Detector.

Por ser este el tipo sensor utilizado, se pasa a realizar una descripción más profunda a continuación.

El **Sensor de Bucle Inductivo (Induction Loop) o E1 Detector** es capaz de detectar vehículos que pasan por su "zona de control" (idealmente una recta contenida en el plano de la calzada y perpendicular a la dirección en la que se mueven los vehículos) y generar salidas que aportan información acerca de estos vehículos como la velocidad media de los vehículos que lo han cruzado, el número de vehículos que han cruzado su zona a controlar o el porcentaje de tiempo que ha estado ocupado, es decir, el porcentaje de tiempo en el que un vehículo ha estado ocupando la "zona de control" anteriormente mencionada.

Para la realización de este proyecto solamente se han utilizado sensores de este tipo por dos razones. Por un lado, el hecho de que se encuentren en una etapa de desarrollo más avanzada respecto a los otros dos tipos de sensores que se pueden utilizar en SUMO nos garantiza una menor tasa de fallos (de hecho, es el único que funciona correctamente vía TraCI), además, para la implantación del sistema en una red real, se trata de los sensores más económicos y abundantes.

En la Figura 5 se puede ver el aspecto de estos sensores durante una simulación.

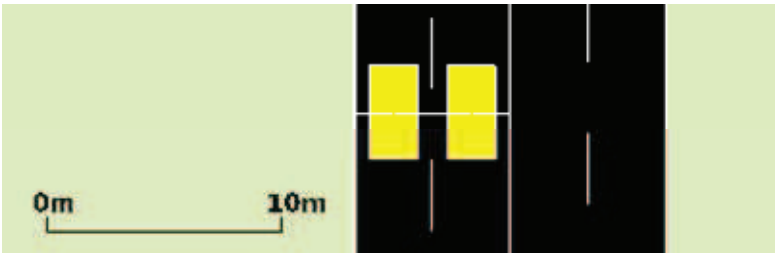


Ilustración 5. Detalle de un sensor de tipo E1

3.2 Semáforos

SUMO permite la simulación de escenarios con cruces controlados por semáforos. En caso de que los cruces no estén controlados por estos, los vehículos esperan para cruzar a que los vehículos provenientes de su derecha hayan abandonado el cruce. En caso de que los cruces si estén controlados por semáforos los estados de estos se pueden fijar de dos formas.

Se puede, o bien crear un programa TLS (Traffic Light Simulation) que defina las fases de una duración determinada que se repetirán cíclicamente, o bien asignar el estado a los semáforos de forma dinámica a través de TraCI.

Siempre que creamos una red a través de la aplicación netconvert.exe, de la que hablaremos en el siguiente apartado, SUMO le asigna automáticamente un programa TLS a todos aquellos cruces que definamos como controlados por semáforos a través de un algoritmo incluido en esta aplicación, pero, para llevar a cabo el proyecto, se interactuará con los semáforos vía TraCI imponiendo la estrategia de control óptima obtenida frente a la asignada automáticamente por SUMO.

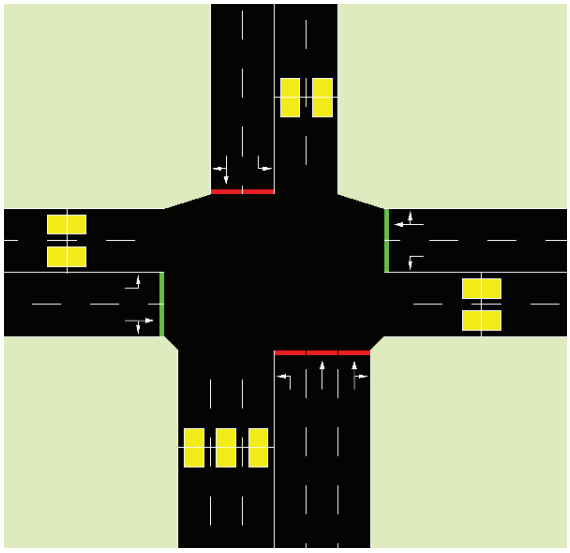


Ilustración 6. Detalle de un cruce semaforizado

3.3 Aplicaciones ejecutables

SUMO está diseñado en forma de diferentes aplicaciones ejecutables que realizan tareas específicas. Las de mayor importancia para este proyecto son tres netconvert.exe, duarouter.exe y sumo-gui.exe.

NETCONVERT.EXE es la aplicación encargada de construir un escenario fijo en el que tendrá lugar la simulación, es decir, esta aplicación creará la red de calles en un formato en que SUMO sea capaz de interpretarla (un archivo *.net.xml) a partir de uno o varios archivos de entrada.

Los inputs a partir de los cuales se generó la red en el proyecto anterior consisten en cuatro ficheros de entrada. Uno de ellos define la posición de los cruces (*.nod.xml), un segundo fichero especifica las calles que los conectan (*.edg.xml), otro define los tipos de vía (*.typ.xml) y un último archivo aporta la información de las posibles conexiones en los cruces, es decir, las posibles direcciones que se pueden tomar en un cruce si se entra a este desde un determinado carril (*.con.xml).

En el proyecto realizado este año, se utilizó también la aplicación netconvert.exe, pero la red se generó a partir de otro tipo de entradas. El proceso se detallará en el Capítulo 3 de esta memoria.

DUAROUTER.EXE es la aplicación encargada de generar un archivo con información referente a las rutas de forma que SUMO pueda simularlas. Esta aplicación requiere como entrada un fichero (*.trip.xml) que contiene información sobre las calles origen y destino de cada trayecto, así como del instante de tiempo en que el vehículo que realizará la ruta debe iniciar su recorrido. La aplicación, con esta información, y utilizando el archivo de salida generado por netconvert (*.net.xml) genera un fichero en el que se define la ruta que debe de seguir cada vehículo (*.rou.xml). En este archivo se indican, en el orden en el que el vehículo ha de pasar por ellas, todas las calles por las que el vehículo ha de pasar para completar su trayecto siguiendo la ruta más corta.

SUMO-GUI.EXE es la aplicación encargada de ejecutar la simulación y de mostrar gráficamente el desarrollo de esta. Para poder ejecutarla se necesitan como entradas varios archivos, en uno de ellos estará contenida la información relativa al escenario fijo o red (*net.xml), un segundo fichero contendrá información relativa a los escenarios variables o rutas (*.rou.xml) y dos archivos más contendrán información referente a las características de la simulación (*sumo.cfg y *.set.xml). Además, puede ser necesario, como en el caso de este proyecto, la utilización de archivos adicionales como los que indican al simulador la presencia de sensores, especificándole cuantos, de que tipo y con qué frecuencia de lectura (*.det.xml).

4 Escenarios

Como se expuso en la introducción de este Capítulo, durante el curso pasado solo se llegó a ejecutar una optimización cuyos parámetros actualizados en tiempo real eran el número de vehículos entrantes por el contorno de la zona a controlar.

Esta optimización se llevó a cabo en un escenario compuesto por seis cruces, todos ellos semaforizados. La sensorización del escenario, al estar destinada a medir únicamente el número de vehículos entrantes a la zona a controlar, se compone de sensores de tipo E1 (mencionados en el apartado 3 del presente Capítulo) situados en todos los carriles entrantes al escenario.

Un detalle del Escenario Sencillo se puede ver en la Figura 7.

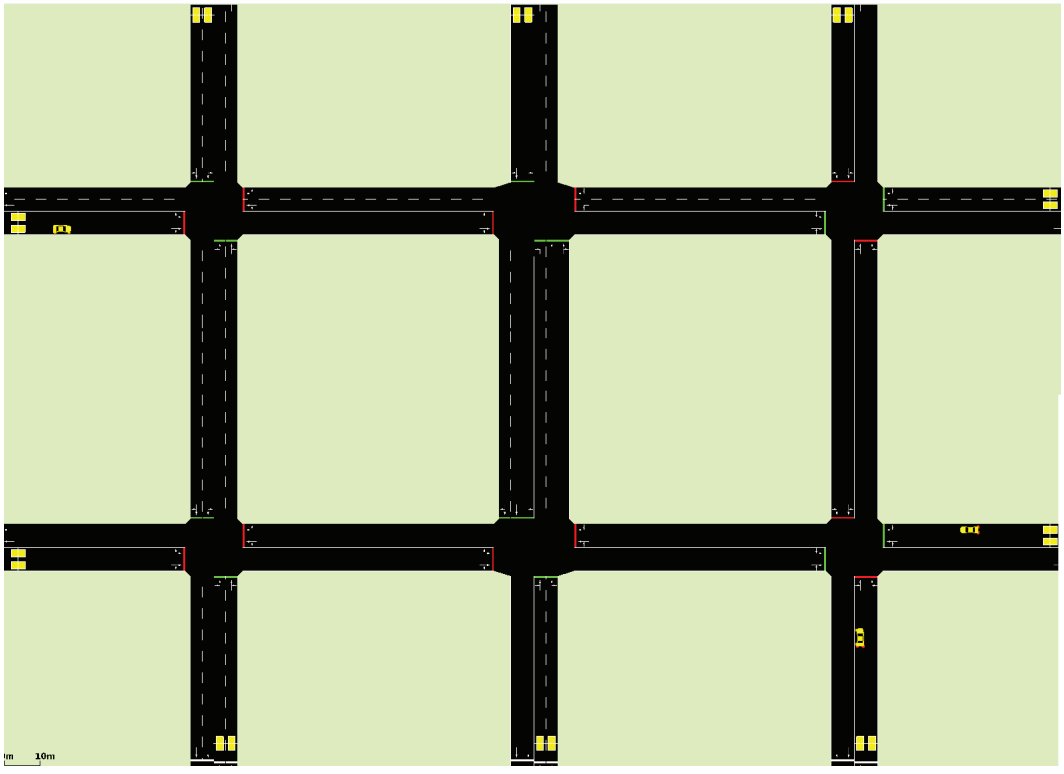


Ilustración 7. Escenario Sencillo

5 Sistema de Control

En el caso del sistema de control, el desarrollado durante el proyecto anterior cubría las necesidades existentes para llevar a cabo la optimización planteada.

El sistema fue desarrollado en Python debido a las necesidades descritas anteriormente. Una parte de este sistema fue tomada como base para desarrollar el sistema de control realizado durante este proyecto.

El sistema de control desarrollado durante el curso pasado era capaz de:

-Lanzar y coordinar la ejecución del simulador tomando el control de este.

-Medir el número de vehículos entrantes en la zona a controlar.

-Ejecutar la optimización y enviar al optimizador el fichero en el que se encontraban los parámetros necesarios para que se pudiese resolver el modelo (número de vehículos entrantes en la zona a controlar).

-Actualizar los valores del número de vehículos entrantes por el contorno.

-Cambiar el estado de los semáforos de acuerdo a la estrategia obtenida tras ejecutar la optimización.

6 Modelo matemático

El modelo matemático programado durante el pasado curso, obtenía una estrategia óptima de control de semáforos en el escenario sencillo.

Este modelo constaba de unos parámetros fijos indicativos de las características de la red y los ratios de giro en cada cruce. Además, utilizaba un parámetro variable encargado de representar el número de vehículos entrantes en la zona a controlar.

Si bien este modelo se utilizó como base para el aprendizaje y la programación del modelo utilizado este año, hubo que realizar en él un gran número de cambios que permitieron que fuese posible llevar a cabo la optimización en escenarios de mayor tamaño y complejidad teniendo en cuenta un mayor número de parámetros tomados en tiempo real. Estos cambios, así como el modelo utilizado este año se detallarán en el Capítulo 4.

7 Generador de escenarios

Durante el proyecto anterior, se programó una interfaz de usuario que permitió que la utilización de la arquitectura diseñada se pudiese llevar a cabo de forma sencilla e intuitiva para realizar pruebas.

Esta interfaz se ha utilizado este año realizando pequeñas modificaciones sobre ella relativas a nombres de archivos de salida, nombres de las calles de origen y destino de las rutas,…

Los cambios necesarios para utilizar esta interfaz en los nuevos escenarios fueron mínimos.

La interfaz de usuario diseñada, se puede considerar que consta de dos módulos.

El primero consiste en una serie de aplicaciones programadas en VBA para Microsoft Excel, gracias a las que, a través de la introducción de unos parámetros sencillos como densidad general del tráfico, rutas específicas (definidas por las calles de origen y destino y unos parámetros

estadísticos que se corresponderán con los de la función según la cual se generan los vehículos que seguirán esa ruta) y tiempos de inicio y fin tanto de la actividad en las rutas específicas como de la simulación, se obtienen unos ficheros de salida que sintetizan esta información para que sea procesada por el segundo módulo de la Interfaz.

Este segundo módulo consiste en un programa escrito en Python que es llamado desde las aplicaciones VBA del primer módulo. Su función consiste en, una vez recopilados los distintos datos generados por las aplicaciones en VBA en base a los requisitos introducidos por el usuario, generar una serie de archivos, tanto de configuración como de rutas, de forma que SUMO se capaz de interpretarlos y llevar a cabo la simulación.

Capítulo 3 ESTADO ACTUAL DEL PROYECTO

Este capítulo tiene como propósito explicar cuáles han sido los desarrollos realizados en este proyecto y su alcance.

Para ello se expondrá primero, de forma genérica, el alcance del proyecto realizado. A continuación se profundizará en la arquitectura previamente expuesta en el Capítulo 1. Tras esto se tratarán en profundidad el sistema de control desarrollado, el modelo matemático y los escenarios sobre los que se ha trabajado. Por último se tratará como se realizaron las transformaciones de mapas a través de las que se consigue la automatización del proceso de implantación en nuevos escenarios.

1 Introducción

En este proyecto se desarrolló un sistema de gestión inteligente del tráfico aplicable a un escenario lo suficientemente grande y complejo como una ciudad.

Para ello hubo que diseñar una arquitectura que permitiese la comunicación entre el simulador y un agente de control a través de la cual el sistema de control obtiene los ratios de giro en los cruces y el número de vehículos entrantes por las diferentes zonas del contorno. Estos datos son suficientes para que el optimizador, cuya ejecución controla el sistema de control, sea capaz de interpretar la situación del tráfico en tiempo real y obtener una configuración de semáforos adecuada al mismo a través de la resolución de un modelo para ser utilizada en el escenario en el que tiene lugar la simulación.

Por otro lado, para favorecer la implantación y las pruebas del sistema en otros escenarios distintos a los estudiados, se añadieron a la arquitectura dos nuevos módulos. Uno de ellos está constituido por OpenStreetMap, un proveedor de mapas de código libre que nos permite extraer fácilmente el mapa de cualquier ciudad o zona en la que nos interese probar el sistema. Después de extraer el mapa, tras una pequeña transformación en SUMO que convierte el mapa en apto para su uso en simulaciones, se realizan en este archivo distintas transformaciones utilizando la herramienta InfoSphere DataStage a través de las cuales se obtienen de forma automática los parámetros necesarios para el optimizador, que representan la posición de calles y semáforos en la red así como los archivos de colocación de los sensores necesarios y otro tipo de datos que se utilizarán en la configuración del sistema de control para el escenario en cuestión.

2 Arquitectura

La arquitectura de este proyecto consta de seis módulos diferenciados como se ve en la Figura 8. Cuatro de ellos atañen al sistema de gestión desarrollado en sí mismo y los otros dos tienen como función la implantación del sistema en nuevos escenarios de forma automática.

Entre los cuatro primeros módulos se encuentran el simulador, el sistema de control, la interfaz de usuario y el optimizador.

El primer paso para la puesta en marcha de la simulación atañe a la interfaz de usuario. En esta se han de especificar las condiciones bajo las que se requiere la ejecución de la simulación en cuanto a niveles y densidad del tráfico, así, se pueden introducir en esta valores de

"densidad de tráfico base", que se verán reflejados en un tráfico variable en cuanto a orígenes y destinos o valores de tráfico para rutas concretas.

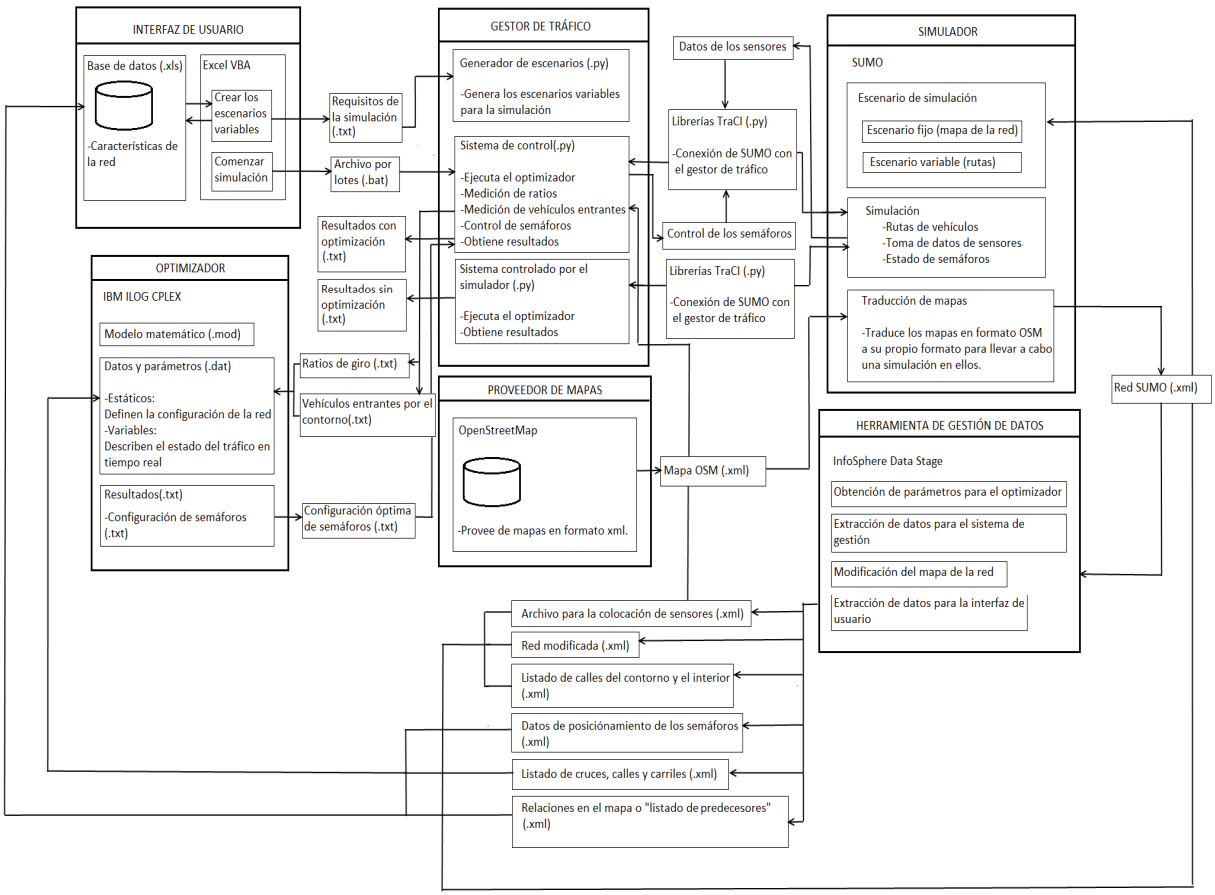

INTERFAZ DE USUARIO
Base de datos (.xls)
-Características de la red
Excel VBA
Crear los escenarios variables
Comenzar simulación
Requisitos de la simulación (.txt)
Archivo por lotes (.bat)
GESTOR DE TRÁFICO
Generador de escenarios (.py)
-Genera los escenarios variables para la simulación
Sistema de control(.py)
-Ejecuta el optimizador
-Medición de ratios
-Medición de vehículos entrantes
-Control de semáforos
-Obtiene resultados
Sistema controlado por el simulador (.py)
-Ejecuta el optimizador
-Obtiene resultados
Datos de los sensores
Librerías TraCI (.py)
-Conexión de SUMO con el gestor de tráfico
Control de los semáforos
Librerías TraCI (.py)
-Conexión de SUMO con el gestor de tráfico
SIMULADOR
SUMO
Escenario de simulación
Escenario fijo (mapa de la red)
Escenario variable (rutas)
Simulación
-Rutas de vehículos
-Toma de datos de sensores
-Estado de semáforos
Traducción de mapas
-Traduce los mapas en formato OSM a su propio formato para llevar a cabo una simulación en ellos.
Red SUMO (.xml)
Resultados con optimización (.txt)
Resultados sin optimización (.txt)
OPTIMIZADOR
IBM ILOG CPLEX
Modelo matemático (.mod)
Datos y parámetros (.dat)
-Estáticos:
Definen la configuración de la red
-Variables:
Describen el estado del tráfico en tiempo real
Resultados(.txt)
-Configuración de semáforos (.txt)
Ratios de giro (.txt)
Vehículos entrantes por el contorno(.txt)
Configuración óptima de semáforos (.txt)
PROVEEDOR DE MAPAS
OpenStreetMap
-Provee de mapas en formato xml.
Mapa OSM (.xml)
HERRAMIENTA DE GESTIÓN DE DATOS
InfoSphere Data Stage
Obtención de parámetros para el optimizador
Extracción de datos para el sistema de gestión
Modificación del mapa de la red
Extracción de datos para la interfaz de usuario
Archivo para la colocación de sensores (.xml)
Red modificada (.xml)
Listado de calles del contorno y el interior (.xml)
Datos de posiciónamiento de los semáforos (.xml)
Listado de cruces, calles y carriles (.xml)
Relaciones en el mapa o "listado de predecesores" (.xml)

Ilustración 8. Arquitectura detallada del proyecto

La especificación de valores de tráfico para rutas concretas se podrá utilizar para comprobar la funcionalidad del sistema ante eventos multitudinarios o accidentes, especificando en la interfaz el valor del flujo vehicular que esta ruta tendrá (en vehículos por minuto) y su origen y destino.

Una vez que el usuario ha especificado las condiciones de tráfico para la simulación, estas son transferidas al sistema de control en el que se

transforman para que SUMO (el simulador) pueda hacer efectivas estas rutas especificadas.

Cuando el simulador tiene disponibles tanto los archivos que definen las rutas como el mapa de la red en la que se llevará a cabo la simulación, esta puede comenzar.

Durante la simulación, el sistema de control y el simulador estarán comunicados de forma continua a través de las librerías de TraCI, enviando el sistema de control los estados de los semáforos al simulador y el simulador las lecturas de los sensores al sistema de control.

Pese a tratarse de una comunicación continua, el sistema de control recopilará esta información cada cinco minutos, y es entonces cuando la tratará para traducir las lecturas de los sensores en ratios de giro y número de vehículos entrantes por el contorno de forma que estos datos se puedan incluir en el modelo. Tras esto, dará la orden de ejecución de la optimización, y una vez que esta acabe recogerá los resultados en forma de estados óptimos de los semáforos para poder implantar la estrategia óptima en el escenario simulado.

Se utilizaron también en la arquitectura otros dos módulos para la implantación rápida del sistema en cualquier escenario. Estos son un proveedor de mapas online, OpenStreetMap y una herramienta de transformación de datos, InfoSphere DataStage.

Para la implantación de sistema en cualquier otro escenario, lo primero que se necesita es exportar el mapa de la zona a controlar en formato *.xml desde la página principal de OpenStreetMap.

Una vez que el mapa ha sido extraído, es necesario traducirlo al formato *.net.xml en el que SUMO define sus mapas, para ello, se utiliza la aplicación netconvert.exe, incluida en el propio simulador.

Cuando se tiene el mapa en el formato en el que se pueden llevar a cabo simulaciones sobre él, es necesario extraer de este datos que reflejen las conexiones entre calles y la situación de los semáforos, así como asignar índices a cada calle y cruce de forma que el optimizador pueda llevar a cabo la simulación en base a estos. Dada esta nueva reorganización de las calles en índices con un nuevo orden, también será necesario adecuar la estructura del mapa a este nuevo orden, como se verá más adelante en el Capítulo 5.

Por último, para poder realizar la simulación, se obtendrán a partir del mapa tanto un archivo para la colocación de los sensores como una indicación de la localización de estos y la de los semáforos de cara a incluir esta información al sistema de control.

3 Sistema de Control

Como ya se comentó en los Capítulos 1 y 2, para la realización del proyecto fue necesaria la realización de un nuevo sistema de control que permitiese el funcionamiento del sistema en tiempo real así como en escenarios reales, grandes y complejos, con cruces tanto gestionados por semáforos como sin semaforizar.

Así, fue necesario desarrollar un sistema de control que permitiese alcanzar los objetivos propuestos, para lo que fue necesaria la realización de distintas tareas.

La primera de todas ellas consistió en la **distinción entre cruces gestionados por semáforos y cruces sin semáforos**, y si bien esta información se obtuvo del mapa a través de las transformaciones de datos realizados con InfoSphere DataStage, fue necesaria la inclusión de esta

distinción en el sistema de control, para poder tratar a cada cruce de acuerdo a sus características.

Un segundo requisito que un sistema de control debe de cumplir para poder funcionar en un escenario real es la **distinción entre el número de posibles direcciones que existen en cada cruce**, ya que en un escenario real los cruces tendrán distintos números de calles conectados a él. En nuestro caso, para realizar la mencionada distinción, se decidió incluir esta información en los identificadores a los sensores, facilitando así al sistema de control la información a través de la lectura de un vector en el que se incluyen los distintos números de identificación de los sensores de bucle inductivo utilizados. Así, pongamos el caso, cuando el sistema de control lee en este vector la información de un sensor que identifica como "00001000040" conoce automáticamente que se trata de un sensor que está colocado a la salida del cruce uno, concretamente en la dirección definida con el índice cuatro.

Para conseguir la **obtención de la información en tiempo real del estado del tráfico**, el sistema de control obtiene dos tipos de datos. El primero es el número de vehículos entrantes por el contorno y el segundo los ratios de giro que están teniendo lugar en cada cruce. Los datos se obtienen a través de la lectura de los sensores cada escalón de tiempo que el sistema de control ordena simular a SUMO, teniendo estos escalones una duración de 100ms de tiempo de simulación. El número de vehículos entrantes por cada calle del contorno y los ratios de giro en cada cruce son los datos mínimos necesarios para que el optimizador pueda adaptar el modelo a través de sus parámetros actualizados cada cinco minutos a la situación actual del tráfico, tanto en cuanto a nivel de congestión como en cuanto a rutas que los vehículos están siguiendo. Para esto, es necesario

que el sistema de control divida los sensores en dos grupos de cara a obtener la información organizada en torno a estos.

Un primer grupo lo constituyen los sensores situados a la salida de los cruces que se encuentran en el interior de la zona a controlar, estas lecturas serán utilizadas para la obtención de los ratios en tiempo real.

Un segundo grupo lo constituyen los sensores situados en el contorno del escenario a través de los cuales se obtiene el número de vehículos que está entrando por cada calle del contorno, información que, combinada con los ratios de giro, indicará el número de vehículos que circulan por cada calle del escenario.

Una vez que el sistema obtiene toda esta información ha de organizarla en matrices cada cinco minutos e incluirla en un fichero de texto de cara a que la optimización se ejecute en base a esta.

Es necesario también que los sensores que envían información al sistema de control, más allá de contar el número de vehículos que cruzan su zona a controlar, puedan **detectar rápidamente una aglomeración en una de las colas.** Aglomeración que, mediante el simple sistema de cuenta de vehículos no serían capaces de detectar si la optimización previa a la aglomeración indicada hubiese fijado una fracción de verde muy limitada para el semáforo en el que se está produciendo la cola, reduciendo el número de vehículos que pueden abandonar esta, y por tanto, el número de vehículos que cruzan la zona de control del sensor.

Así, se optó por añadir la variable de ocupación al sistema de control, de forma que, ante grandes colas sin liberación de estas por parte del semáforo, un elevado incremento de la ocupación del sensor producirá el mismo efecto que un gran número de vehículos atravesando la zona de control del sensor indicado.

Por último, también es necesario que el **sistema de control sea capaz de establecer el estado de los semáforos** obtenido como resultado de la optimización.

Tras la puesta en marcha del sistema con las características citadas anteriormente, fue necesario realizar dos correcciones, la primera atañe a errores en las medidas de los sensores y la segunda a problemas en el establecimiento del estado óptimo de los semáforos.

La primera corrección realizada atañe a los errores de medida que presentaban los sensores inductivos del simulador, estos se deben a la diferencia de velocidad en los vehículos que los cruzan, dándose el caso de que, ante velocidades de ejecución de la simulación elevadas, si un vehículo cruza la zona de control de un sensor a una velocidad baja, será contabilizado un número de veces mayor que un vehículo que lo cruce a una velocidad alta.

Para esto, se añadió al sistema de control la extracción de datos relativos a la velocidad media de los vehículos que cruzan cada sensor y se modificaron las medidas del número de vehículos multiplicando este por un factor dependiente de la velocidad. Tras varias iteraciones, se tomó como factor de corrección:

$$K(v) = \frac{v\left[\frac{m}{s}\right]}{9}$$

Ecuación 1. Factor de corrección k dependiente de la velocidad.

Siendo v la velocidad media de los vehículos que atraviesan un detector concreto en el último escalón de tiempo en m/s.

La segunda y última corrección necesaria trata del establecimiento del estado óptimo de los semáforos, ya que, cuando el sistema de control fija el estado de los semáforos mientras se ejecuta la simulación a velocidades elevadas, surgen errores en la lectura del vector de estados de semáforos que provocan el fallo de SUMO. Para solucionar este problema, se incluyó en el bucle de fijación del estado de los semáforos la instrucción `try-continue`, a través de la cual, ante un error en el proceso de fijación del estado de los semáforos, se mantiene el estado anterior hasta el siguiente escalón de tiempo, en el que se fijará el estado que corresponda.

4 Modelo Matemático

Para la realización de este proyecto, fue necesario el desarrollo de un modelo matemático que tuviese en cuenta la situación del tráfico en tiempo real, asimismo, este modelo debe de ser los suficientemente rápido y eficaz como para poder trabajar en escenarios grandes y complejos como una ciudad y realizar las optimizaciones en un tiempo lo suficientemente corto como para no comprometer la funcionalidad del sistema.

El modelo matemático se expondrá de forma más detallada en el Capítulo 4 de la presente memoria, pero se pretende en este apartado dar una visión general de su funcionamiento, y cuáles son las especificaciones que este debió de cumplir para poder ser utilizado en el proyecto.

El modelo matemático debe de cumplir con lo siguiente:

- Ser fácilmente instalable en un nuevo escenario, es decir, se requiere que sea lo suficientemente sencilla la modificación de los parámetros utilizados para describir el escenario sobre el cual se lleva a cabo la optimización.

- Tener en cuenta el estado del tráfico en tiempo real, tanto en lo que atañe a nivel de este (número de vehículos en el escenario) como a su comportamiento (ratios de giro en cada cruce).
- Ser capaz de trabajar tanto sobre cruces semaforizados como sin semaforizar.
- Ser lo suficientemente rápido como para no comprometer la fiabilidad y robustez del sistema (se ha fijado como tiempo límite para la optimización tres minutos).
- Generar en forma de archivos de salida el estado óptimo de los semáforos y los niveles de tráfico esperados en cada cruce y calle.

5 Escenarios y extracción de datos

Para la realización de este proyecto se programaron distintas fases de desarrollo a través de las cuales se pudo ir obteniendo información y experiencia de cara a llegar al conocimiento total de toda la arquitectura para poder implantar esta en cualquier escenario.

Así, se implantó el sistema de control en tres escenarios distintos, aumentando paulatinamente el nivel de complejidad de estos, como se ve en la Tabla 1. Gracias a esto, se pudo lidiar con las mayores dificultades en las etapas más tempranas del proyecto, en las que el escenario era más sencillo y los errores en el sistema más fácilmente identificables y resolubles.

Los tres escenarios fueron denominados Escenario Sencillo, Escenario Parcial y Barrio Completo. El nivel de complejidad los distintos escenarios se muestra en la Tabla 1.

	Escenario Sencillo	*Escenario Parcial*	*Barrio completo*
Número de cruces	6	30	328
Número de cruces semaforizados	6	24	97

Tabla 1. Nivel de complejidad de los escenarios

El Escenario Sencillo consta de seis cruces, todos ellos semaforizados, y constituye la red donde se realizarán las pruebas y se desarrollará el sistema. Para facilitar esta tarea, todos los cruces tienen el mismo número de calles conectadas a ellos y las calles discurrirán a lo largo del escenario en solo dos direcciones distintas, asimismo, las longitudes de todas las calles serán similares como se puede apreciar en la Figura 9.

El Escenario Parcial representa una parte del Barrio de la Misericordia compuesta por treinta cruces, veinticuatro de ellos semaforizados, como se puede observar en la Figura 10. Este escenario se pudo utilizar para validar lo aprendido en la implementación del sistema de control en el Escenario Sencillo a la vez que constituyó un escenario perfecto para el estudio, aprendizaje e implantación del método de automatización de la creación de escenarios y la extracción de datos.

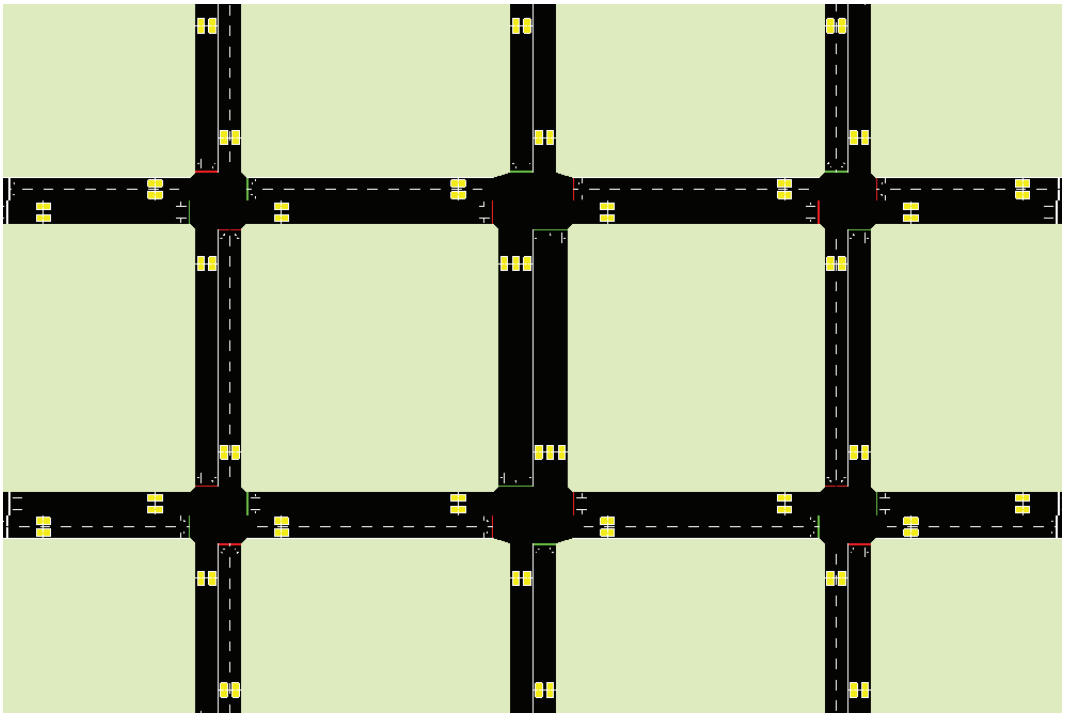


Ilustración 9. Escenario Sencillo sensorizado

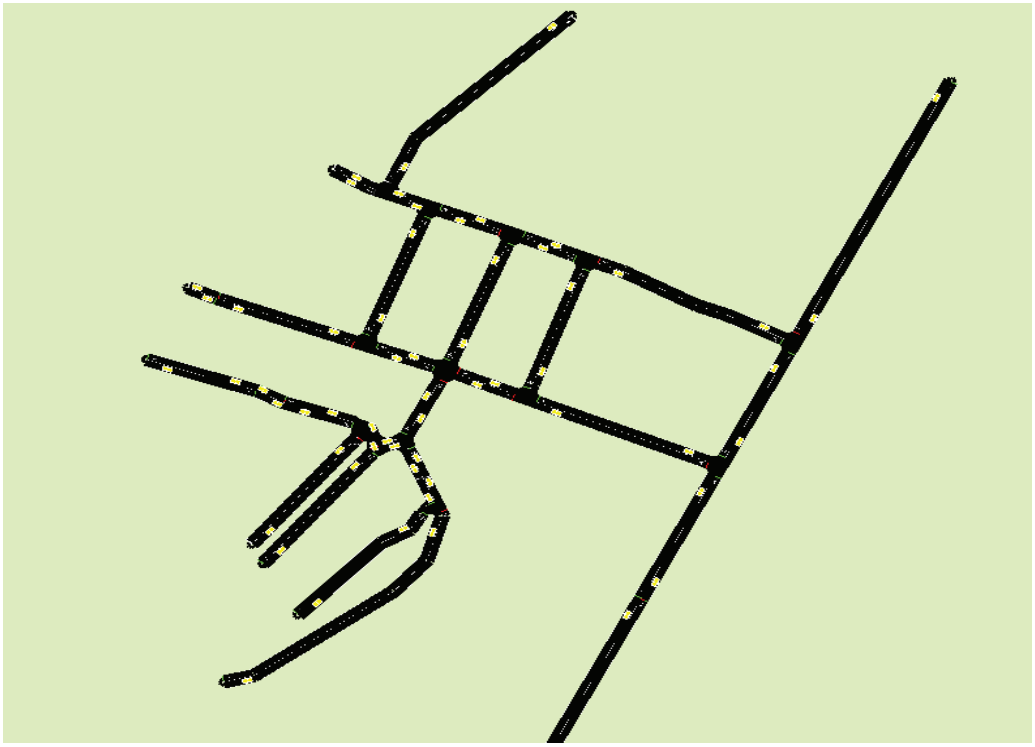


Ilustración 10. Escenario Parcial

Por último, se comprobó el funcionamiento tanto del método para automatizar la implantación del sistema como del sistema de control en un escenario más complejo como el Barrio de la Misericordia, que se puede observar en la Figura 11.

Ilustración 11. Escenario correspondiente al Barrio Completo

Capítulo 4 MODELO MATEMÁTICO

En este capítulo se expondrá el modelo matemático utilizado para la obtención de la configuración óptima de los semáforos de acuerdo al estado del tráfico.

Se tratarán en primer lugar los índices asignados, para pasar a continuación a definir los parámetros utilizados, las variables de decisión, la función objetivo y las restricciones que modelan el comportamiento del tráfico respectivamente

1 Introducción

Antes de introducir el modelo, es necesario realizar una serie de consideraciones preliminares.

La primera de todas ellas se refiere a como se considera en el modelo el número de coches que atraviesan una vía. Se ha de decir que los vehículos se tratan en este modelo través de los flujos vehiculares que conforman.

La segunda consideración se refiere a la discretización del tiempo, en este modelo se consideran 120 escalones de tiempo (o ciclos) de 5 segundos, siendo esta la unidad más pequeña de tiempo contemplada. Las implicaciones de esto se verán en que los cambios en el estado de los semáforos solo se pueden producir en un instante de tiempo múltiplo del tiempo del ciclo.

2 Índices

El modelo presentado es aplicable a un escenario con un número de cruces cualquiera y cualquier número de vías que acceden a cada uno de estos cruces. Para definir ambos elementos se han utilizado los índices del modelo. Para establecer estos índices a la par que se agiliza la aplicación del modelo a un nuevo escenario, se tendrá en cuenta el cruce con un mayor número de vías conectadas con él, y se asignará este número de posibles vías a todos los cruces, así, los cruces que tengan menos "direcciones" aparecerán en el modelo como si tuviesen las mismas que el cruce con mayor número de conexiones, pero todas las vías que sobre el escenario no existan en estos cruces tendrán flujos vehiculares nulos, no conectarán con ningún otro cruce y no tendrán semáforos, por lo que no influirán en el modelo.

Así, utilizaremos un índice i para referirnos a un cruce concreto, y un índice j para referirnos a un carril de un cruce. Al tener distintos cruces una dirección j dada, ha de tenerse en cuenta que para tratar un carril concreto se ha de dar un índice j asociado a un cruce con un índice i, por lo que se tratará como "j(i)".

$$i = \text{Índice que representa un cruce concreto}$$

Ecuación 2. Índice "i".

$$j = \text{Índice que representa la vía por la que se accede a un cruce}$$

Ecuación 3. Índice "j".

$$t = \text{Índice que representa un ciclo de tiempo}$$

Ecuación 4. Índice "t".

$$c = [i, j] = \text{Índice que representa la } combinación \text{ de un cruce "i" y un carril "j" que llega a él}$$

Ecuación 5. Índice "c"

Se han de diferenciar a continuación los carriles que forman parte del contorno de la zona a controlar (a los que accederían sin pasar por ningún cruce intermedio coches provenientes del exterior del escenario) de los carriles pertenecientes al interior del escenario.

$$L_{Contorno} = \text{Subespacio que representa el conjunto de "c" que forman parte del contorno}$$

Ecuación 6. Subespacio "$L_{Contorno}$".

$$L_{Interior} = \text{Subespacio que representa el conjunto de "c" que forman parte del interior}$$

Ecuación 7. Subespacio "$L_{Interior}$".

También se han de diferenciar los carriles en función de su dirección, de forma que en un cruce no se pueda permitir nunca la circulación proveniente de dos carriles por los que se circule en distintas direcciones. Sí se podrá permitir la circulación simultánea por un cruce de vehículos provenientes de dos carriles con la misma dirección y sentidos contrarios. Para hacer esta diferencia entre carriles con distintas direcciones se definen dos subespacios cuya intersección será nula y cuya suma representará el total de los carriles del escenario.

$$L_{Dir1} = \text{Subespacio que representa el conjunto de carriles c que llegan a los diferentes cruces en dirección 1}$$

Ecuación 8. Subespacio "L_{Dir1}".

$$L_{Dir2} = \text{Subespacio que representa el conjunto de carriles c que llegan a los diferentes cruces en dirección 2}$$

Ecuación 9. Subespacio "L_{Dir2}".

Se requiere también para la optimización de un subespacio para cada elemento del conjunto "c" que trate de sus posibles conexiones, es decir, que refleje la realidad del mapa que se está simulando. Para ello, se le asignará a cada elemento un subespacio en el que se incluirá el conjunto de elementos de "c" desde los cuales se puede acceder al elemento tratado.

$$L_{Predecesores}(c) = \text{Subespacio que representa el conjunto de "c" que son predecesores del "c" tratado}$$

Ecuación 10. Subespacio "$L_{Predecesores}(c)$".

Por último, y debido a la asignación mencionada anteriormente de un número de carriles a cada cruce igual para todos pese a sus diferencias, se establecerán tres subespacios, que reflejarán los carriles que tienen predecesores, cuáles de estos forman partes del interior, y que carriles pertenecientes al interior no están conectados con ningún otro, que son aquellos que no se han de tener en cuenta para la optimización ya que no existen realmente en el escenario.

$L_{CarrilesConPredec}$ = Subespacio que representa el conjunto de carriles "c" que tienen alguna conexión

Ecuación 11. Subespacio "$L_{CarrilesConPredec}$".

$L_{InteriorPredecesores}$ = Subespacio que representa el conjunto de "c" que pertenecen simultáneamente a $L_{CarrilesConPredec}$ y a $L_{Interior}$

Ecuación 12. Subespacio "$L_{InteriorPredecesores}$".

$L_{InteriorSinPredecesores}$ = Subespacio que representa el conjunto de "c" que pertenecen a $L_{Interior}$ sin pertenecer a $L_{CarrilesConPredecesores}$

Ecuación 13. Subespacio "$L_{InteriorSinPredecesores}$".

3 Parámetros

Como se puede ver en la Figura 8, este modelo maneja dos tipos de parámetros: aquellos que no varían entre una optimización y otra, y aquellos que si lo hacen.

Dentro del primer grupo, parámetros estáticos, se encuentran aquellos que hacen referencia a las características de la red, y que, por tanto, tienen siempre el mismo valor.

En el segundo grupo se encuentran los parámetros variables, que hacen referencia al estado del tráfico y son actualizados cada vez que se

ejecuta la simulación, así, en este grupo se encuentran los ratios de giro (*"Ratios(c)"*) y el número de vehículos entrantes por cada carril del contorno (*"EntrantesContorno(c)"*).

$$NbSemaforos(c) = Número\ de\ semáforos\ que\ gestionan\ el\ carril\ "c"$$

Ecuación 14. Parámetro *"NbSemaforos(c)"*

$$NbSemaforosCruce(i) = Número\ total\ de\ semáforos\ que\ gestionan\ los\ distintos\ carriles\ del\ cruce\ "i"$$

Ecuación 15. Parámetro "$NbSemaforosCruce(i)$"

$$EntrantesContorno(c) = Flujo\ vehicular\ que\ recorre\ el\ carril\ "c"$$

Ecuación 16. Parámetro "$EntrantesContorno(c)$"

Nótese que pese a ser solamente de utilidad el valor de vehículos entrantes por el contorno en los carriles del propio contorno, este parámetro se ha definido para todos los carriles. Así, en los carriles que no pertenezcan al contorno, su valor será cero, y además, no se utilizará en las restricciones, como se verá en la sección 4.5 del presente Capítulo.

$$FV = Número\ máximo\ de\ vehículos\ que\ pueden\ salir\ de\ un\ carril\ en\ un\ ciclo\ de\ tiempo$$

Ecuación 17. Parámetro *"FV"*

$$T_{periodo} = Número\ de\ ciclos\ que\ definen\ un\ periodo\ de\ tiempo$$

Ecuación 18. Parámetro *"Tperiodo"*

$$T_{minV} = Número\ \ mínimo\ de\ ciclos\ que\ debe\ estar\ un\ semáforo\ en\ verde\ en\ \ un\ periodo\ de\ tiempo\ T_{periodo}$$

Ecuación 19. Parámetro *"TminV"*

$$T_{SemSteady} = Número\ mínimo\ de\ ciclos\ que\ debe\ permanecer\ en\ verde\ un\ semáforo\ si\ se\ pusiera\ en\ verde$$

Ecuación 20. Parámetro *"TSemSteady"*

$$T_{desp} = Número\ de\ ciclos\ que\ tarda\ un\ vehículo\ en\ desplazarse\ de\ un\ carril\ a\ otro\ tras\ un\ semáforo$$

Ecuación 21. Parámetro *"Tdesp"*

$$NumGrande = Numero\ grande$$

Ecuación 22. Parámetro *"NumGrande"*

$$Ratios(c) = Porcentaje\ de\ vehículos\ que\ van\ al\ carril\ c\ una\ vez\ han\ llegado\ al\ cruce\ al\ que\ este\ pertenece$$

Ecuación 23. Parámetro *"Ratios(c)"*

4 Variables

Para el llevar a cabo una optimización el optimizador necesita de ciertas variables interrelacionadas entre sí por restricciones y a las que puede fijar su valor. Así, en el modelo vendrán descritas variables como el

estado de los semáforos, que es la variable a través de la cual se influye en el estado tráfico, o las colas en los distintos cruces, que son la variable a minimizar. Para llegar a establecer el valor de una variable (las colas) en función de la otra (los semáforos) el optimizador utiliza también numerosas variables de apoyo como los flujos entrantes y salientes en cada carril, variables binarias que indican en qué dirección se abren los semáforos de un determinado cruce, etc.

Todas las variables de este modelo son positivas.

Las variables del modelo son las siguientes:

$$Sem(t,c) = Variable\ binaria\ que\ indica\ si\ el\ semáforo\ del\ carril\ "c"\ se\ encuentra\ en\ verde\ en\ el\ ciclo\ "t"$$

Ecuación 24. Variable *"Sem(t,c)"*

$$Colas(t,c) = Número\ de\ vehículos\ que\ se\ encuentran\ en\ el\ carril\ "c"\ en\ el\ ciclo\ "t"$$

Ecuación 25. Variable *"Colas(t,c)"*

$$FlujoSalMax(t,c) = Número\ máximo\ de\ vehículos\ que\ podrían\ salir\ del\ carril\ "c"\ en\ el\ ciclo\ "t"$$

Ecuación 26. Variable *"FlujoSalMax(t,c)"*

$$FlujoSal(t,c) = Número\ de\ vehículos\ que\ salen\ del\ carril\ "c"\ en\ el\ ciclo\ "t"$$

Ecuación 27. Variable *"FlujoSal(t,c)"*

$$FlujoEnt(t,c) = Número\ de\ vehículos\ que\ entran\ al\ carril\ "c"\ en\ el\ ciclo\ "t"$$

Ecuación 28. Variable *"FlujoEnt(t,c)"*

$$AlgunV_1(t,i) = Variable\ binaria\ que\ indica\ si\ algún\ semáforo\ del\ cruce\ "i"\ que\ habilita\ el\ tráfico\ en\ la\ dirección\ 1\ se\ encuentra\ en\ verde\ en\ el\ ciclo\ de\ tiempo\ t$$

Ecuación 29. Variable *"AlgunV1(t,i)"*

$$AlgunV_2(t,i) = Variable\ binaria\ que\ indica\ si\ algún\ semáforo\ del\ cruce\ "i"\ que\ habilita\ el\ tráfico\ en\ la\ dirección\ 2\ se\ encuentra\ en\ verde\ en\ el\ ciclo\ de\ tiempo\ t$$

Ecuación 30. Variable *"AlgunV2(t,i)"*

5 Función objetivo

La estrategia de optimización en base a la que se buscará el estado de semáforos óptimo consiste en una minimización de las colas en la red, consiguiendo así una minimización del tiempo de permanencia de los vehículos en la red, una reducción del número de vehículos en la red y una reducción de las emisiones de estos.

$$\min \sum_{t,c} Colas(t,c)$$

Ecuación 31. Función Objetivo

6 Restricciones

Las restricciones aquí expuestas tienen como finalidad relacionar entre sí las distintas variables de decisión sobre las que el optimizador puede actuar. A través de estas restricciones se conseguirá que el optimizador tenga en cuenta la realidad del tráfico y sus características, así como las distintas condiciones necesarias para garantizar la seguridad en la asignación de semáforos.

Las primeras restricciones a tener en cuenta son aquellas que definen las colas para un instante t como las colas previamente existentes más el balance neto de vehículos que se suman a la cola en ese carril.

Asimismo, se ha de tener en cuenta que las colas han de ser siempre positivas.

$$Colas(t,c) \geq Colas(t-1,c) + FlujoEnt(t,c) - FlujoSalMax(t,c)$$
$$\forall\, t \geq 1, c$$

Ecuación 32. Definición de las colas (1)

$$Colas(t,c) = Colas(t-1,c) + FlujoEnt(t,c) - FlujoSal(t,c)$$
$$\forall\, t \geq 1, c$$

Ecuación 33. Definición de las colas (2)

$$Colas(t,c) \geq 0$$
$$\forall\, t \geq 1, c$$

Ecuación 34. Definición de las colas (3)

Como ya se ha comentado en la sección 4.4 de este capítulo, el objetivo de la optimización llevada a cabo no es otro que el de reducir el número total de colas, así, serán estas las que determinen el estado de los semáforos.

Teniendo en cuenta esto y observando las restricciones anteriormente expuestas, podemos darnos cuenta de que el valor de los semáforos para el primer ciclo de tiempo no tendrá ninguna implicación en los valores de estas colas, y por tanto, será indiferente para el simulador a la hora de obtener la estrategia de control óptima, así, se ha decidido fijar este valor para todos los semáforos igual al que tendrán en el segundo ciclo de tiempo a través de la siguiente restricción.

$$Sem(0,c) = Sem(1,c) \quad \forall\, c$$

Ecuación 35. Asignación de *"Sem(0,c)"*

Sabiendo que los valores de las colas, variable a minimizar, dependen de los flujos de entrada y salida debemos definir estos dos en función de otras variables.

Para definir los flujos de salida, lo primero será definir el flujo de salida máximo que puede tener lugar en cada carril y que se define de la siguiente forma:

Para carriles gestionados por semáforos, en caso de que el semáforo esté en verde el flujo de salida máximo se corresponderá con el número de vehículos que pueden salir de la cola por unidad de tiempo y en caso de que el semáforo se encuentre en rojo, su valor será nulo.

Para carriles que no se encuentran gestionados por semáforos, se considerará un valor del flujo de salida máximo igual al número de vehículos que pueden salir de la cola por unidad de tiempo.

$$FlujoSalMax(t,c) \leq FV \cdot Sem(t,c)$$
$$\forall\, t,c \Rightarrow NbSemaforos(c) > 0$$

Ecuación 36. Valor de *"FlujoSalMax(t,c)"* para carriles controlados por semáforos

$$FlujoSalMax(t,c) \leq FV$$
$$\forall\, t,c \Rightarrow NbSemaforos(c) = 0$$

Ecuación 37. Valor de *"FlujoSalMax(t,c)"* para carriles no controlados por semáforos

Por otro lado, para los flujos entrantes en una cola concreta, se distinguirán dos casos:

En el caso de que el carril tratado forme parte del contorno, este flujo entrante estará determinado por las lecturas de los sensores en tiempo real, las cuales se definen según el parámetro "*EntrantesContorno*" que se actualiza en cada optimización.

En el caso de que el carril tratado forme parte del interior del escenario, el flujo entrante será igual al número de vehículos que están accediendo al cruce que es predecesor de nuestro carril multiplicado por la probabilidad de que esos vehículos se dirijan efectivamente a la cola objeto de estudio.

$$FlujoEnt(t,c) \geq \frac{EntrantesCotorno(c)}{NbCiclos}$$
$$\forall\, t,c \in L_{Contorno}$$

Ecuación 38. Valor de *"FlujoEnt(t,c)"* si "c" pertenece al contorno

$$FlujoEnt(t,c) \geq Ratios[i,j] \cdot \sum_{k} FlujoSal\left(t - T_{desp}, [i,k]\right)$$

$$\forall\, t \geq T_{desp}, c \in L_{InteriorPredecesores}, [i,j] \in L_{Predecesores}(c)$$

Ecuación 39. Valor de *"FlujoEnt(t,c)"* si "c" pertenece al interior

También se han de utilizar dos restricciones para realizar ciertos ajustes en los flujos entrantes. El primer ajuste atañe a que para tiempos transcurridos desde el inicio de tiempos considerado en la la optimización menores al que los vehículos necesitan para moverse de un carril a otro, todos los flujos entrantes en los carriles que conforman el interior del escenario serán nulos. El segundo ajuste alivia capacidad de computación al optimizador, fijando a cero los valores de flujo entrante para aquellos carriles del interior que no tiene predecesores, es decir, para aquellos carriles que aparecen definidos de cara a la implantación más rápida del sistema en un nuevo escenario pero que no se corresponden con ningún carril del mapa.

$$FlujoEnt(t,c) = 0$$

$$\forall\, t < T_{desp}, c \in L_{Interior}$$

Ecuación 40. Valor de *"FlujoEnt(t,c)"* si "c" es del interior y "t" es menor que *"Tdesp"*

$$FlujoEnt(t,c) = 0$$

$$\forall\, t \geq T_{desp}, c \in L_{InteriorSinPredecesores}$$

Ecuación 41. Valor de *"FlujoEnt(t,c)"* si "c" es del interior y no tiene predecesores

También se han añadido al modelo las restricciones necesarias para que la estrategia de semáforos sea segura, limitando la apertura de semáforos en los cruces a una sola dirección a la vez.

$$Sem(t,[i,j]) = AlgunV_1(t,i)$$
$$\forall\, t,[i,j] \in L_{Dir1} \Rightarrow NbSemáforos[i,j] > 0$$

Ecuación 42. Establecimiento del valor de *"Sem(t,c)"* para los semáforos en dirección 1.

$$Sem(t,[i,j]) = AlgunV_2(t,i)$$
$$\forall\, t,[i,j] \in L_{Dir2} \Rightarrow NbSemáforos[i,j] > 0$$

Ecuación 43. Establecimiento del valor de *"Sem(t,c)"* para los semáforos en dirección 2.

$$AlgunV_1(t,i) + AlgunV_2(t,i) = 1$$
$$\forall\, t,i \Rightarrow NbSemaforosCruce(i) \geq 0$$

Ecuación 44. Restricción de seguridad para los cruces

En este modelo no aparecen representados los peatones, por lo cual se establecerá un tiempo mínimo que todo semáforo debe permanecer en verde en un periodo de tiempo dado.

$$\sum_{tt=t-T_{periodo}}^{t} Sem(tt,c) \geq T_{minV}$$
$$\forall\, t \geq T_{periodo}, c$$

Ecuación 45. Restricción de tiempo mínimo en verde

Por otro lado, para completar la descripción de la realidad de la gestión de semáforos, se añadió una restricción más que evita que los semáforos cambien de estado de forma intermitente.

$$\sum_{tt=t-(T_{SemSteady}-n)}^{tt=t+(T_{SemSteady}-n)} Sem(tt,c) \geq (T_{SemSteady}+1-n)\cdot Sem(t,c)$$

$$\forall\, c, 0 < n < T_{SemSteady}$$

$$\forall\ T_{SemSteady}-n \leq t \leq NbCiclos - T_{SemSteady} + n - 1$$

Ecuación 46. Restricción para la continuidad del semáforo en verde

Pese a que con estas restricciones el tráfico quedaría perfectamente modelado se hizo necesario añadir dos restricciones más debido a las características de los mapas extraídos de OpenStreetMap.

Cuando se extrae un mapa de este proveedor de mapas y se convierte a un formato apto para que SUMO realice en él una simulación, es probable que en las calles del contorno aparezcan semáforos en una sola dirección en cruces que el simulador considera como tales pese a no serlo.

Si se diese la condición de que el número de vehículos circulando por estas calles indebidamente semaforizadas por el simulador fuese cero, sería indiferente para el simulador fijar un estado u otro, pero resulta evidente la conveniencia de fijar su estado permanentemente en verde de cara a los posibles cambios que se puedan dar en el estado del tráfico, así, esta condición queda recogida en las dos siguientes restricciones.

$$\sum_{j \Rightarrow [i,j] \in L_{Dir1}} NbSemaforos\,([i,j]) \leq NbSemaforosCruce(i) - 1 + NumGrande * AlgunV1(t,i)$$

$$\forall\, t,i \Rightarrow NbSemáforosCruce(i) > 0$$

Ecuación 47. Identificación y fijación de estado de los semáforos de los cruces que solo tienen semáforos solo en la dirección 1

$$\sum_{j \Rightarrow [i,j] \in L_{Dir2}} NbSemaforos\,([i,j]) \leq NbSemaforosCruce(i) - 1 + NumGrande * AlgunV2(t,i)$$

$$\forall\, t,i \Rightarrow NbSemáforosCruce(i) > 0$$

Ecuación 48. Identificación y fijación de estado de los semáforos de los cruces que solo tienen semáforos solo en la dirección 2

7 Scripts de postprocesamiento

Una vez que se haya resuelto la optimización, se deben crear dos documentos de texto.

El primero se generará para que la información relativa al estado óptimo de los semáforos sea recogida en un fichero cuyo formato sea interpretable por el sistema de control y así este pueda establecer el estado óptimo de los semáforos.

El documento que este script generará contendrá solo ceros y unos que el sistema de control transformará en estados en rojo y verde respectivamente, además, el sistema de control incluirá en todos los pasos de verde a rojo una cantidad determinada de tiempo en ámbar.

El segundo documento se generará de forma que se recoja en este la información relativa a las colas en las diferentes calles. Este documento se

genera con la finalidad de que esa información sea recogida por un agente externo al proyecto para mostrársela al usuario, así, este podrá conocer el estado del tráfico en las diferentes vías y decidir que ruta utilizará para llegar a su destino teniendo en cuenta esta información.

Capítulo 5 AUTOMATIZACIÓN DEL PROCESO DE IMPLANTACIÓN DEL SISTEMA

Este capítulo tiene como propósito explicar el proceso de implantación del sistema en nuevos escenarios y como este fue automatizado a través del uso del proveedor de mapas online OpenStreetMap y la herramienta de gestión de información InfoSphere DataStage.

En la primera sección de este capítulo se detallará el proceso a través del cual se puede obtener rápidamente un nuevo mapa para SUMO de la zona en la que hayamos decidido implementar nuestro sistema de control.

La segunda sección detalla la información que es necesario obtener o incluir en el mapa, así como el proceso a través del cual esta se obtiene automáticamente.

1 Automatización de la creación de escenarios

Para conseguir que el sistema de control desarrollado se convirtiese en una herramienta realmente útil y aliviar la carga de trabajo que supone la implantación de este en un nuevo escenario, se decidió plasmar el conocimiento adquirido acerca de la creación de nuevos escenarios y la implantación en estos de nuestro sistema en un proceso automático de cara a que no fuese necesario nada más allá de la lectura del manual de usuario incluido en la Parte II de esta Memoria.

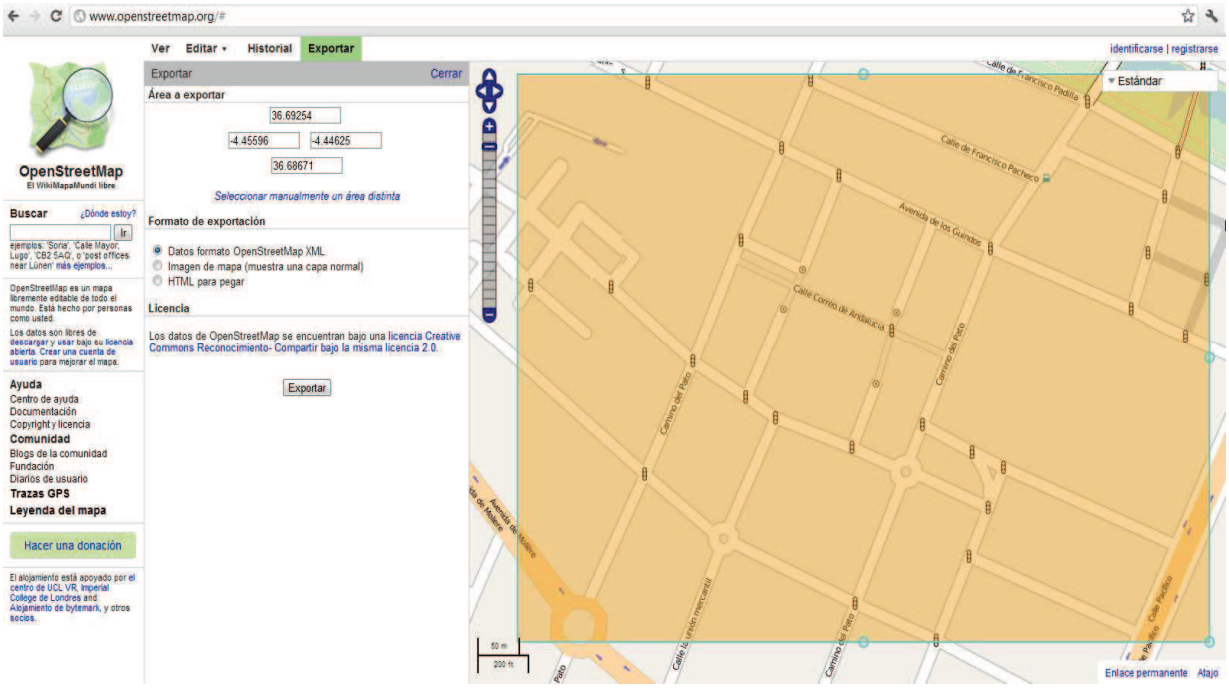


Ilustración 12. Detalle de la extracción de un mapa desde OpenStreetMap

La automatización del proceso de creación de escenarios se inicia con la extracción del mapa de la zona a controlar en formato *.xml del proveedor de mapas online OpenStreetMap. Lo primero que se ha de hacer antes de exportarlo es comprobar la semaforización de la zona. En caso de los semáforos no se encuentren incluidos en el mapa debemos de añadirlos nosotros en el propio OpenStreetMap, tratando de reproducir con la mayor fidelidad posible la colocación de semáforos en el área en la que vayamos a llevar a cabo nuestra simulación.

Una vez este mapa ha sido extraído, se le aplicarán las transformaciones necesarias para convertirlo en un mapa cuyo formato acepte SUMO para realizar simulaciones sobre él. Para esto utilizaremos la aplicación netconvert.exe que se incluye en SUMO, y transformaremos el archivo escribiendo lo siguiente en la línea de comandos:

```
netconvert --osm-files map osm.xml -o map.net.xml
```

sustituyendo "map" por el nombre del fichero que queremos transformar.

Una vez se haya realizado esta transformación, ya tenemos preparado el mapa para que en él se pueda probar nuestro sistema de control, pero de cara a la implantación de este será necesario reflejar la realidad física de la red en los distintos módulos que conforman la arquitectura a través de la inclusión de ciertos parámetros.

En la siguiente sección se describen los parámetros necesarios y como se automatizó su obtención.

2 Parámetros necesarios y método de obtención

En este apartado se describirá la etapa de implantación del sistema de control desarrollado en un nuevo escenario, para ello, se describirá primeramente el archivo que tomaremos como partida para realizar las transformaciones. Este no es otro que el mapa extraído de OpenStreetMap ya traducido al formato apto para realizar en él simulaciones con SUMO. En la segunda parte se detallarán los parámetros que es necesario extraer de este mapa de cara a implantar en el nuestro sistema de control y a que parte del archivo de entrada corresponde la información a extraer.

Por último, se detallará el proceso a través del cual se han extraído estos datos y realizado las transformaciones.

2.1 Especificaciones para la extracción de datos

2.1.1 Especificación del documento XML de entrada

La entrada será un archivo XML de estructura similar al que se muestra en el ejemplo expuesto en la sección 2.2.2 del presente capítulo. El parecido es meramente estructural, ya que el archivo que se usa como ejemplo es mucho más corto y no cumple ciertas condiciones que sí cumple el XML completo, es decir, se ha copiado un trozo de cada parte del archivo (se ha copiado una parte de los "edges", una parte de los "junctions", una parte de los "tl-logic"… estando al menos un ejemplo o más de todas las partes, pero todas incompletas) pudiendo darse incongruencias en este ejemplo, como el caso en que, cuando se define una calle ("edge") con sus etiquetas "from: junction salida" y "to: junction final", luego no se encuentren los "junction" (cruces) de salida o llegada en la parte que atañe a la definición de los "junction".

Esto es fruto solo del recorte que se ha realizado al archivo y no sucede en un mapa real.

Lo que el archivo de partida especifica es lo siguiente (el ejemplo coloreado para una mejor comprensión se encuentra en el apartado 2.1.2 del presente capítulo):

Existe una primera parte que no contiene información que interesante para las transformaciones, y está en color verde.

En una segunda parte, se definen una serie de calles ("edges") que son internos y para uso exclusivo del simulador, estas calles simplemente permiten o imposibilitan el giro hacia ciertas direcciones en un cruce (vienen a ser algo así como las condiciones de "prohibido girar a la izquierda" pero dadas de forma inversa, ya que una de estas calles permite tomar una salida determinada en un cruce para una entrada

dada). Esta información tampoco es interesante para las transformaciones. En el ejemplo está coloreada de rojo.

Para diferenciar las calles internas al simulador, que no interesan a la hora de realizar las transformaciones, de las que conforman el mapa como tal, basta con comprobar el valor de la etiqueta "function" que aparece en la descripción de la calle, así, las que tengan un valor "function=internal" no serán calles que conformen el mapa, y su análisis no es de interés, mientras que las que tengan un valor de "function" distinto de "internal" sí son calles a tener en consideración para las transformaciones, su definición está en color azul en el ejemplo y se definen con los siguientes atributos:

```
<edge id="69701300#1" from="833615328" to="833615330" priority="4"
type="highway.residential" function="normal">
<lanes>
<lane id="69701300#1_0" depart="1" maxspeed="13.89" length="11.69"
shape="58.43,429.69 48.02,435.03"/>
</lanes>
</edge>
```

Así, en este caso la calle ("edge") tiene un identificador "id=69701300#1", y se puede identificar en el mapa porque va del cruce "833615328" (etiqueta "from") al "833615330" (etiqueta "to"). Esta calle tiene un solo carril, ya que solo existe una definición dentro de "lanes", y este tiene una longitud de "11.69" (etiqueta "length" dentro de "lane").

A continuación se definen en el XML los "row-logic", que en el ejemplo se ven en color marrón, su definición tendrá la siguiente forma:

```
<row-logic id="833615335" requestSize="9" laneNumber="3">
<logic>
<logicitem request="0" response="000000000" foes="100010000"
cont="0"/>
<logicitem request="1" response="011000000" foes="011110000"
cont="1"/>
<logicitem request="2" response="010001000" foes="010001000"
cont="1"/>
<logicitem request="3" response="010000000" foes="010000100"
cont="0"/>
<logicitem request="4" response="010000011" foes="110000011"
cont="1"/>
<logicitem request="5" response="001000010" foes="001000010"
cont="1"/>
<logicitem request="6" response="000000000" foes="000100010"
cont="0"/>
<logicitem request="7" response="000000000" foes="000011110"
cont="0"/>
<logicitem request="8" response="000010001" foes="000010001"
cont="1"/>
</logic>
</row-logic>
```

Se ha de recalcar aquí que el id de un "row-logic" coincide con el del cruce al que está referido, y que el "row-logic" es la parte que especifica al simulador las instrucciones acerca de cómo se deben de comportar o como deben de interactuar los coches al llegar a un cruce. La información de los "row-logic" no es interesante para las transformaciones. La única información importante asociada a esta parte del documento es que hay "row-logic" con un solo "logicitem" definido, y otros, como el de arriba, que tienen varios "logicitem" asociados.

Solo los cruces cuyo "row-logic" tenga varios "logicitem" asociados, podrán tener un "tl-logic" asociado.

Los tl-logic se definen a continuación de los "row-logic", en el ejemplo figuran en color violeta:

```
<tl-logic id="248605200" type="static" programID="0" offset="0">
<phase duration="31" state="G"/>
<phase duration="4" state="y"/>
</tl-logic>
```

De la misma forma que los "row-logic", los "tl-logic" están asociados a un cruce ("junction") en concreto, y se conoce porque el campo "id" del "tl-logic" coincidirá con el campo "id" del cruce al que está asociado, de aquí se puede extraer la información de que cruces están semaforizados y cuáles no. Estarán semaforizados aquellos cuya id aparezca como id en un tl-logic.

A continuación aparece en el XML la definición de los "junction" o cruces, estos también pueden ser de dos tipos atendiendo al valor de su atributo "type", los primeros, cuyo "type" es diferente a "internal", son los que proporcionarán información de interés y en el ejemplo figuran en color amarillo. Los cruces de interés o "junctions" cuyo "type" es distinto de internal se definen de la siguiente forma:

```
<junction id="833615330" type="priority" x="47.05" y="433.68"
incLanes="69701300#1_0 -69701300#2_0" intLanes=":833615330_0_0 :833615330_4_0
:833615330_2_0 :833615330_5_0" shape="48.75,436.45 45.79,430.67 47.82,436.84"/>
```

En este caso el "id" del "junction" es "id=833615330", por lo que si estuviese semaforizado encontraríamos en la parte del documento en la

que se definen los "tl-logic" un "tl-logic" con ese id. Su "type" es "type=priority", por lo que es de interés para realizar las transformaciones al no ser de tipo "type=internal", como los que se definirán más adelante. El cruce está georeferenciado con "x=47.05 y=433.68" y se definen las calles entrantes a ese cruce a través de una enumeración de sus carriles entrantes en la etiqueta incLanes="69701300#1_0 -69701300#2_0".

Otro tipo de cruces de interés, no considerados "internal" son los que tienen como valor de type "type=DEAD_END", ya que estos son los cruces pertenecientes al contorno del mapa, los cruces que marcan el fin del mapa.

A continuación, en color rosa en el ejemplo aparecen definidos una serie de cruces cuyo valor de la etiqueta "type=internal" nos da a entender que se trata de datos que usa el simulador para apoyar en ellos su funcionamiento "interno" y que no son de interés.

Por último se definen en negro, distintas variables de las calles y carriles que no se habían definido previamente:

```
<succ edge="69701295" lane="69701295_0" junction="833615337">
<succlane lane="69701296_0" via=":833615337_6_0" tl="833615337"
linkno="6" yield="1" dir="r" state="t"/>
<succlane lane="69701299_0" via=":833615337_7_0" tl="833615337"
linkno="7" yield="1" dir="l" state="t" int_end="x"/>
<succlane lane="-69701295_0" via=":833615337_8_0" tl="833615337"
linkno="8" yield="1" dir="t" state="t" int_end="x"/>
</succ>
```

Así, en "succ edge…" se da una combinación de calle "69701295", carril "69701295_0", y cruce junction="833615337", y a continuación trata las conexiones que se ofrecerán para acceder a las distintas vías a las que el

mapa le permite acceder (constituyendo el resto de calles del cruce aquellas a las que el giro desde el carril tratado no está permitido) definidas por separado como "succlane". El atributo más importante de estas se podría considerar el valor de "tl=833615337" ya que, mientras sea un número significará que la intersección está semaforizada, mientras que si su valor es tal que "tl="" significará que no existe un semáforo para tal desplazamiento de un carril a otro a través de un cruce.

Existe otro parámetro importante en esta definición, el "linkno" asociado a cada semáforo concreto (entendiendo que cada semáforo controla la posibilidad o no de moverse hacia un solo carril de destino, y que si un semáforo controla la posibilidad de moverse hacia dos carriles de destino se considerará entonces como dos semáforos).

Así, este "linkno" indica la posición del vector que define el estado de los semáforos del cruce que indicará el estado concreto del semáforo tratado.

2.1.2 Ejemplo de un mapa de OpenStreetMap

```
<?xml version="1.0"?>
<!-- generated on 03/31/12 18:28:14 by SUMO netconvert Version 0.12.1
<?xml version="1.0" encoding="iso-8859-1"?>
<configuration>
<input>
<osm-files value="map.osm"/>
</input>
<output>
<output-file value="misericordia.net.xml"/>
</output>
<projection>
<proj.utm value="true"/>
```

```
</projection>
</configuration>
-->
<net>
<location netOffset="-370424.04,-4061661.35"
convBoundary="0.00,0.00,340.49,618.72"          origBoundary="-4.450499,36.691784,-
4.446712,36.697379"  projParameter="+proj=utm  +zone=30  +ellps=WGS84  +datum=WGS84
+units=m +no_defs"/>
<edge id=":1145423208_0" function="internal">
<lanes>
<lane     id=":1145423208_0_0"     depart="0"     maxspeed="13.89"     length="4.82"
shape="79.68,544.24 78.19,544.13 77.43,543.62 77.38,542.70 78.05,541.37"/>
</lanes>
</edge>
<edge id=":833615205_12" function="internal">
<lanes>
<lane     id=":833615205_12_0"     depart="0"     maxspeed="13.89"     length="4.72"
shape="186.04,494.79 186.02,494.89 185.39,495.57 184.48,495.47 183.27,494.60"/>
</lanes>
</edge>
<edge id=":833615205_6" function="internal">
<lanes>
<lane     id=":833615205_6_0"     depart="0"     maxspeed="13.89"     length="5.05"
shape="181.54,498.71 182.71,498.02 183.39,497.11 183.58,495.97 183.27,494.60"/>
</lanes>
</edge>
<edge id=":833615205_7" function="internal">
<lanes>
<lane     id=":833615205_7_0"     depart="0"     maxspeed="13.89"     length="9.48"
shape="181.54,498.71 190.36,495.24"/>
</lanes>
</edge>
<edge     id="69701300#1"     from="833615328"     to="833615330"     priority="4"
type="highway.residential" function="normal">
<lanes>
<lane     id="69701300#1_0"     depart="1"     maxspeed="13.89"     length="11.69"
shape="58.43,429.69 48.02,435.03"/>
</lanes>
</edge>
```

```
<edge id="69701300#2" from="833615330" to="833615222" priority="4"
type="highway.residential" function="normal">
<lanes>
<lane id="69701300#2_0" depart="1" maxspeed="13.89" length="49.27"
shape="47.32,435.32 0.52,450.73"/>
</lanes>
</edge>
<edge id="99006875" from="1145423337" to="1145423260" priority="5"
type="highway.unclassified" function="normal">
<lanes>
<lane id="99006875_0" depart="1" maxspeed="22.22" length="113.62"
shape="105.11,535.08 115.46,557.23 166.54,604.09 181.18,617.50"/>
</lanes>
</edge>
<row-logic id="833615334" requestSize="1" laneNumber="1">
<logic>
<logicitem request="0" response="0" foes="0" cont="0"/>
</logic>
</row-logic>
<row-logic id="833615335" requestSize="9" laneNumber="3">
<logic>
<logicitem request="0" response="000000000" foes="100010000" cont="0"/>
<logicitem request="1" response="011000000" foes="011110000" cont="1"/>
<logicitem request="2" response="010001000" foes="010001000" cont="1"/>
<logicitem request="3" response="010000000" foes="010000100" cont="0"/>
<logicitem request="4" response="010000011" foes="110000011" cont="1"/>
<logicitem request="5" response="001000010" foes="001000010" cont="1"/>
<logicitem request="6" response="000000000" foes="000100010" cont="0"/>
<logicitem request="7" response="000000000" foes="000011110" cont="0"/>
<logicitem request="8" response="000010001" foes="000010001" cont="1"/>
</logic>
</row-logic>
<row-logic id="833615337" requestSize="9" laneNumber="3">
<logic>
<logicitem request="0" response="000000000" foes="100010000" cont="0"/>
<logicitem request="1" response="000000000" foes="011110000" cont="0"/>
<logicitem request="2" response="010001000" foes="010001000" cont="1"/>
<logicitem request="3" response="000000000" foes="010000100" cont="0"/>
<logicitem request="4" response="000000011" foes="110000011" cont="0"/>
```

<logicitem request="5" response="001000010" foes="001000010" cont="1"/>
<logicitem request="6" response="000000010" foes="000100010" cont="0"/>
<logicitem request="7" response="000011010" foes="000011110" cont="1"/>
<logicitem request="8" response="000010001" foes="000010001" cont="1"/>
</logic>
</row-logic>
<tl-logic id="248605200" type="static" programID="0" offset="0">
<phase duration="31" state="G"/>
<phase duration="4" state="y"/>
</tl-logic>
<tl-logic id="248605290" type="static" programID="0" offset="0">
<phase duration="31" state="G"/>
<phase duration="4" state="y"/>
</tl-logic>
<junction id="833615330" type="priority" x="47.05" y="433.68" incLanes="69701300#1_0 -69701300#2_0" intLanes=":833615330_0_0 :833615330_4_0 :833615330_2_0 :833615330_5_0" shape="48.75,436.45 45.79,430.67 47.82,436.84"/>
<junction id="833615334" type="priority" x="44.68" y="358.18" incLanes="-69701294_0" intLanes=":833615334_0_0" shape="44.72,358.15 47.16,356.08 42.20,360.28 44.64,358.21"/>
<junction id="833615335" type="priority" x="93.86" y="416.24" incLanes="-69701295_0 69701294_0 -69701300#0_0" intLanes=":833615335_0_0 :833615335_9_0 :833615335_10_0 :833615335_3_0 :833615335_11_0 :833615335_12_0 :833615335_6_0 :833615335_7_0 :833615335_13_0" shape="99.53,411.38 93.57,408.78 92.00,409.01 87.04,413.21 86.85,415.13 88.88,421.30"/>
<junction id="833615337" type="priority" x="98.26" y="406.15" incLanes="-69701299_0 -69701296_0 69701295_0" intLanes=":833615337_0_0 :833615337_1_0 :833615337_9_0 :833615337_3_0 :833615337_4_0 :833615337_10_0 :833615337_6_0 :833615337_11_0 :833615337_12_0" shape="101.53,410.50 103.36,404.26 98.46,401.36 93.50,405.56 93.54,408.85 99.50,411.45"/>
<junction id=":677379889_4_0" type="internal" x="122.40" y="165.63" incLanes=":677379889_1_0 27234353#1_0" intLanes=":677379889_2_0" shape=""/>
<junction id=":677379889_5_0" type="internal" x="125.31" y="164.32" incLanes=":677379889_3_0 -27234353#2_0" intLanes=":677379889_0_0" shape=""/>
<junction id=":677394227_4_0" type="internal" x="207.98" y="332.10" incLanes=":677394227_1_0 27234353#2_0" intLanes=":677394227_2_0" shape=""/>
<junction id=":677394227_5_0" type="internal" x="210.89" y="330.79" incLanes=":677394227_3_0 -27234353#3_0" intLanes=":677394227_0_0" shape=""/>
<succ edge="69701296" lane="69701296_0" junction="833615302">
<succlane lane="-69701296_0" via=":833615302_0_0" tl="833615302" linkno="0" yield="1" dir="t" state="t"/>

```
</succ>
<succ edge="69701299" lane="69701299_0" junction="828321360">
<succlane lane="69010485_0" via=":828321360_6_0" tl="828321360" linkno="6"
yield="1" dir="r" state="t"/>
<succlane lane="-69008829_0" via=":828321360_7_0" tl="828321360" linkno="7"
yield="1" dir="l" state="t"/>
<succlane lane="-69701299_0" via=":828321360_8_0" tl="828321360" linkno="8"
yield="1" dir="t" state="t" int_end="x"/>
</succ>
<succ edge=":1145423272_0" lane=":1145423272_0_0" junction="1145423272">
<succlane lane="-69701249#4_0" tl="" linkno="" yield="0" dir="s" state="M"/>
</succ>
<succ edge=":1145423337_0" lane=":1145423337_0_0" junction="1145423337">
<succlane lane="-69701260#0_0" tl="" linkno="" yield="0" dir="s" state="M"/>
</net>
```

2.2 Especificación de los ficheros de salida

Para que el optimizador y el sistema de control puedan llevar a cabo su tarea correctamente se necesita obtener cierta información del mapa en XML previamente descrito de cara a la configuración del sistema. Esta información se extrajo en forma de diferentes listas, matrices y vectores, y el proceso de extracción se encuentra totalmente automatizado.

A continuación se describen los distintos datos extraídos, dando una breve explicación de la necesidad de su extracción y en que parte del fichero XML que describe el mapa se puede encontrar la información necesaria para extraerla.

NOTA: No se tendrán en cuenta para la extracción de esta información ni las calles (“edges”) ni los cruces (“junctions”) de “type=internal”.

2.2.1 Listado de las calles y cruces

Lo primero que se necesita es obtener un listado con todas las calles y cruces del mapa a la vez que se les asignan los índices que se utilizarán en la optimización y que el sistema de control utilizará para identificarlos.

Para ello, lo primero que debe hacerse es asignar los mencionados índices a los cruces y las calles, es decir, numerar los cruces de 1 en adelante (a este número le llamaremos "MiIdCruce") y hacer lo mismo con las calles de cada cruce, es decir, denominar al primer cruce "leído" cruce 1 y a las calles entrantes o salientes en este cruce de la siguiente forma: <1,0>, <1,1>, <1,2>, <1,3>, …(a este número le llamaremos "MiIdCruceCarril").

Para hacer esto y tener constancia de los cambios realizados, se debe recorrer la definición de los diferentes "junctions" (sin recorrer los "internal junctions") o cruces, en amarillo en el ejemplo, asignándoles un número por el orden en que se vaya realizando la lectura empezando por el uno.

A continuación se deben recorrer, para cada cruce etiquetado, todas las calles o "edges" (sin recorrer las "internal edges") definidas al inicio del documento en azul buscando el id (en este caso el identificador que se les ha asignado en el mapa y no nuestro "MiIdCruce") del cruce ("junction") que corresponda en cada momento tanto en las etiquetas "from" como "to", y, marcando la calle como <1,0> para la primera calle encontrada para el cruce 1, <1,1> para la segunda calle encontrada para el cruce 1, …

Además, si dos calles comparten las mismas etiquetas from y to, o las tienen cambiadas, deben contabilizar como una sola, asignándo el número identificador que se asignó a la calle con la que comparte etiquetas, así la salida de estos datos tendrá la siguiente estructura:

```
<junction id="id del junction que aparece en el xml" MyIdCruce="la que nosotros asignamos p.ej:0" x="valor de la coordenada x del junction" y="valor de la coordenada y del junction">
<edge id="id del edge que aparece en el xml" from="valor del from del edge" to="valor del to del edge" MyIdCruceCarril="la que nosotros asignamos p.ej:<0,0>"/>
<edge id="id del edge que aparece en el xml" from="valor del from del edge" to="valor del to del edge" MyIdCruceCarril="la que nosotros asignamos p.ej:<0,1>"/>
</junction>
```

Puede suceder que una misma calle tenga dos "MiIdCruceCarril", una para el cruce del que "sale" y otra para el cruce al que "llega", esto ha de ser así para que el optimizador funcione. Estas relaciones entre "MiIdCruceCarril" se reflejarán en el archivo de salida "Predecesores" que se comentará en la sección 2.2.3.

2.2.2 Número total de cruces y calles

También es necesario tener en otra salida el número de cruces y el número de "MiIdCruceCarril" para cada cruce (que no es lo mismo que el numero de calles, ver 2.2.1) en forma de tabla o matriz para poder introducirlo directamente al optimizador.

2.2.3 Listado de predecesores

Otra de las salidas necesarias consiste en ligar las distintas "MiIdCruceCarril" que pertenecen a una misma calle, así, si la calle X sale del cruce 4 y va al 8 y está numerada respectivamente <4,1> y <8,5>, se necesita que en una salida salgan todos los predecesores separados por "," con la siguiente estructura:

```
<<4,1><8,5>>, …
```

2.2.4 Listado de carriles que van o vienen del contorno

Es necesario, para hacer las distinciones pertinentes en el modelo matemático, como se expuso en el Capítulo 4, distinguir los carriles que conforman el contorno del escenario de aquellos que forman parte del interior.

Para ello, se generará una salida en forma de dos listas, una llamada entrantes, que incluirá todas las calles cuyo origen (etiqueta "from") pertenece a un "junction" de "type=DEAD_END", y una segunda lista, llamada salientes que incluirá todas las calles cuyo destino (etiqueta "to") pertenece a un "junction" de "type=DEAD_END".

En la lista deben aparecer el id de la calle asignado en el propio mapa, su etiqueta "from", su etiqueta "to", y nuestro "MiIdCruceCarril" asignado. Así, las dos listas tendrán la siguiente estructura:

```
<edge id="id del edge que aparece en el xml" from="valor del from
del edge" to="valor del to del edge" MiIdCruceCarril="Valor de
MiIdCruceCarril asignado al realizar la primera lista"/>
```

2.2.5 Listado de calles en dirección 1 y de calles en dirección 2

Este archivo se necesita para conseguir una configuración de semáforos segura que no permita la apertura de semáforos que habiliten el tráfico en dos direcciones "perpendiculares" en un mismo cruce.

Esta salida tendrá forma de dos listas, una para calles en "Dirección 1" y otra para calles en "Dirección 2".

La información relativa a cada calle, es decir, la estructura del archivo de salida, será la misma que la del listado de carriles que van o vienen del contorno, es decir:

```
<edge id="id del edge que aparece en el xml" from="valor del from
del edge" to="valor del to del edge" MyIdCruceCarril="Valor de
MiIdCruceCarril asignado al realizar la primera lista"/>
```

Para realizar la distinción pertinente sobre la dirección de las calles se ha analizado el mapa y se ha comprobado que para la zona a simular las calles pueden tener o bien dirección SE-NO o bien dirección SO-NE. Así podremos realizar la distinción entre ambas direcciones comprobando la pendiente de las calles.

Una calle se considerará que tiene dirección 1 cuando al dividir el "incremento de coordenada x" entre el "incremento de coordenada y" de los cruces señalados por "to" y "from" respectivamente se obtiene un número mayor que cero (pendiente positiva).

Una calle tiene dirección 2 en caso contrario (número obtenido menor que cero o pendiente negativa).

2.2.6 Listado de cruces sin semaforizar

Es necesario, tanto para el sistema de control como para el modelo matemático, distinguir que cruces se encuentran controlados por semáforos y cuáles no lo están.

Así, este archivo de salida consiste en una lista de cruces cuya id no está asociada a ningún "tl-logic" y deberá contener información relativa tanto al id asignado en el XML como al nuevo: "MiIdCruce".

2.2.7 Listado del número de semáforos en cada carril

Esta lista responde a la necesidad de conocer el número de semáforos que controlan las posibles direcciones que se pueden tomar desde cada carril en un cruce para poder fijar sus estados.

El número de semáforos que controlan cada combinación de "edge", "lane" y "junction" se obtiene de la última parte del XML, ver sección 3.1 del presente capítulo, contando el número de "lanes" dentro de la combinación definida de "edge", "lane" y "junction" cuyo atributo "tl" no está vacío (en las que no aparece tl=" ").

La lista tiene la forma:

```
MiIdCruce="Id que asignamos al cruce",
MiIdCruceCarril=<id que asignamos al crucecarril. Si hay dos,
escogeremos el MiIdCruceCarril en el que el campo correspondiente
al id del cruce coincide con el anterior>
lane="El definido por el xml"
N° semáforos="N° semáforos"
```

2.2.8 Matriz semáforos lane

De cara a incluir en el simulador directamente la información extraída relativa al número de semáforos en cada carril, esta se ha de organizar en una matriz. La matriz se ha de construir de forma que cada posición [i,j] tenga un valor determinado por el número de semáforos correspondiente al MiIdCruceCarril <i,j>.

2.2.9 Vector Semáforos Cruce

El último parámetro relativo a la colocación de semáforos en el escenario que el optimizador necesita para hallar la estrategia de gestión óptima es el número de semáforos en cada cruce.

Por tanto, es necesario obtener un vector en el que el valor de la posición "i" del vector representa la suma se valores de todas las columnas " j" que aparecen en la fila "i" de la matriz descrita anteriormente en la sección 3.2.8 del presente capítulo.

2.2.10 Listados de estándar y de contorno

En esta lista se trata de reorganizar la información referente a la lista de carriles que van o vienen del contorno a la que se hizo referencia en el apartado 3.2.4 del presente capítulo. Si bien la lista mencionada anteriormente tiene como finalidad enviar la información al optimizador, es necesario también que el sistema de control tenga acceso a esta información.

Esta lista responde a la necesidad de colocar sensores en todas las calles habiliten la entrada a la zona a controlar o la salida de esta. Una vez esta lista se halla generado, será suficiente con incluirla en la configuración para que los sensores en las zonas indicadas puedan empezar a realizar la toma de datos.

Para esta salida se ha generado una lista en la que cada elemento tiene la siguiente estructura:

```
<e1-detector id="xxxx" lane="yyyy" pos="10" freq="100"
file="detectores.out" friendly_pos="x"/>
```

Para crear las listas se han de leer todas los "lanes" en las que el "type" del "edge" al que pertenecen sea distinto de "internal" y se hará lo siguiente:

Se lee el identificador del "lane" (aparecen todos los "lanes" al principio del XML cuando definen los "edges", en la parte que del mapa ejemplo en azul) y se coloca en yyyy.

Se lee el "MiIdCruceCarril" del "edge" al que pertenece el "lane", como hay dos "MiIdCruceCarril" asociados a cada "edge" (uno para el "junction" "from" y otro para el "junction" "to"), el "MiIdCruceCarril"

leído será el que está asociado al "junction" del "from"(el que tiene como primera cifra el "MiIdCruce" del "junction" asociado al "from").

Se asocia a xxxx un id de la siguiente forma:

Estará compuesto por los números que componen el "MiIdCruceCarril" que está asociado al "junction from" (el que tiene como primer número el "MiIdCruce" del "junction" asociado al "from" del edge tratado), asignando cinco cifras para representar cada número, poniendo tantos ceros a la izquierda como sea necesario para completar las cinco cifras.

A continuación se asigna un número que indica el orden en el que se han procesado distintas "lanes" que comparten el mismo "MiIdCruceCarril" empezando por el cero, de forma que 2 "lanes" pertenecientes al mismo "edge", o lo que es lo mismo, dos carriles pertenecientes a la misma calle, tengan id del sensor colocado en ellos que se diferencian solo en este número.

Así, por ejemplo, si se lee una lane 998726363_0#0 tendríamos:

yyyy=998726363_0#0

Si cuando se lee su "edge" este resulta ser el 998726363, cuyo "MiIdCruceCarril" es <118,4>, siendo el MyIdCruce del cruce de partida 118, y es la segunda vez en el proceso que etiquetamos una lane de este MiIdCruceCarril (es importante hacer esta comprobación por MiIdCruceCarril y no por el id del "edge", ya que se ha visto que dos "edges" distintos pueden compartir el "MiIdCruceCarril" si comparten "from" y "to"), hemos de etiquetar con 0011800004 por el

"MiIdCruceCarril" correspondiente y seguida de un 1 por ser la segunda que nos encontramos (si fuese la primera llevaría un 0) un carril ("lane") correspondiente a esta calles ("edge"), entonces tenemos un:

xxxx=00118000041

La definición del correspondiente elemento de la lista sería entonces:

```
<e1-detector id="00118000041" lane="998726363_0#0" pos="10"
freq="100" file="detectores.out" friendly_pos="x"/>
```

Una vez tenemos esta definición solo queda decidir en cuál de las dos listas debe de ser incluida, esto dependerá del "type" que tenga asociado el "junction" que aparece en la etiqueta "from" del "edge" al cual pertenece el "lane" en cuestión. Es decir, si el "edge" al que pertenece el "lane", tiene en su "from" un "junction" con "type=DEAD_END", se colocará en la lista 2 o lista "contorno". Si, al contrario, el "junction" del "from" es de "type" distinto de DEAD_END, se asociará a la lista 1 o lista "estándar".

La información de qué "junction" está asociado al "from" del "edge" al que pertenece la "lane" se encuentra al principio del XML, de donde realizamos la lectura de las diferentes "lanes".

La información de cuál es el "type" del "junction" descrito anteriormente se encuentra en la parte del XML en la que se enumeran los "junction", en color amarillo en el ejemplo.

2.2.11 Listado de vectores

Para incluir la información relativa a la colocación de los sensores en el sistema de control es necesario ordenar la información relativa a

estos en dos vectores que se incluirán en una lista. Los vectores simplemente tienen asociados a cada una de sus posiciones un id de detector de los que fueron anteriormente definidos como xxxx (p. ej.: 00118000041).

El vector A tiene asignados a sus posiciones los id de la lista 1 o estándar, mientras que el vector B tiene los id de la lista 2 o contorno. Es importante colocar en cada posición el correspondiente id de detector entre comillas, así tendremos un archivo de salida con la siguiente estructura:

```
A=["00118000040", "00118000041", "00118000050", "00118000051",
"00118000052", "00119000000", …]

B=["00018000000", "00018000001", "00018000010", "00018000011",
"00018000012", "00019000000", …]
```

2.2.12 Modificación Red para hacer coincidir los linkno

A través de las extracciones de datos descritas anteriormente, se ha obtenido la información necesaria para que el optimizador lleve a cabo la optimización.

Esta información trata, entre otras cosas, del número de semáforos en cada carril y cada cruce, y al organizar esta hemos indicado implícitamente al optimizador un orden de asignación de semáforos.

El script de postprocesamiento utilizado en nuestro modelo matemático (expuesto en el Capítulo 4), escribe el estado de los semáforos en un fichero *.txt de acuerdo al orden indicado por la matriz explicada en la sección 2.2.8 del presente capítulo en la que se incluye información acerca del número de semáforos en cada carril de la red. Pero como se

expuso en la sección 3.1.1 del presente capítulo, los semáforos ya tienen un orden previamente asignado durante la transformación del mapa extraído de OpenStreetMap que SUMO lleva a cabo.

Así, la última modificación necesaria tiene como objetivo modificar los linkno, que son los indicadores del "orden" de estos semáforos para hacerlo coincidir con el orden que ha sido asignado en la matriz de semáforos expuesta en la sección 2.2.8 del presente capítulo.

Así, esta transformación consiste en modificar la última parte de la red, donde se define como están configurados los semáforos de cada vía.

Esa parte de la red tiene la siguiente estructura:

```
<succ edge="61649822#1" lane="61649822#1_0" junction="768556270">
<succlane lane="61649822#2_0" via=":768556270_6_0" tl="768556270"
linkno="6" yield="1" dir="r" state="t"/>
<succlane lane="61649851_0" via=":768556270_7_0" tl="768556270"
linkno="7" yield="1" dir="l" state="t" int_end="x"/>
<succlane lane="-61649822#1_0" via=":768556270_8_0" tl="768556270"
linkno="8" yield="1" dir="t" state="t" int_end="x"/>
</succ>

<succ edge="61649822#2" lane="61649822#2_0" junction="768556271">
<succlane lane="61649822#3_0" via=":768556271_0_0" tl="768556271"
linkno="0" yield="1" dir="r" state="t"/>
<succlane lane="61649829_0" via=":768556271_1_0" tl="768556271"
linkno="1" yield="1" dir="l" state="t" int_end="x"/>
<succlane lane="-61649822#2_0" via=":768556271_2_0" tl="768556271"
linkno="2" yield="1" dir="t" state="t" int_end="x"/>
</succ>
```

La modificación se realizará en base a la matriz de semáforos mencionada anteriormente, en la que cada posición [i,j] indica el número de semáforos existentes en el cruce i, dirección j, es decir, al estar así construida la matriz, la fila 0 no se corresponde a ningún cruce (ya que el primer "MiIdCruce" existente es el 1), y deberían de ser 0 todos sus

elementos. De hecho, esta fila será posteriormente eliminada antes de introducir la matriz como parámetro en el optimizador.

La transformación a realizar, modificará el atributo definido como "linkno" para cambiar el orden que se asigna a los semáforos. Para ello las operaciones son las siguientes:

Se han de tomar una por una las filas de la matriz, que representan los cruces en el orden determinado por los "MiIdCruce" asignados.

Se lee el valor del número de semáforos del cruce, es decir, la suma de las columnas para esa fila. Si es cero se pasa a la siguiente, pero si es distinto de cero:

> Se lee el primer valor distinto de cero, y se busca en la parte de la red en la que se define la configuración de los semáforos una definición que tenga:
>
> ```
> <succ edge="xxxx" lane="yyyy" junction="zzzz">
> ```
>
> Donde xxxx es el id del "edge" correspondiente al "edge" cuyo "MiIdCruceCarril" sea <i,j>, siendo i y j son los índices del elemento de la matriz que estamos tratando.
>
> Donde zzzz sea el id del "junction" cuyo "MiIdCruce" tenga como valor i.
>
> Una vez encontrado, se modifican los valores de los "linkno" situados dentro de este campo, asignando valores contiguos empezando por el 0. Es decir, si el elemento [i,j] de la matriz era 2, fijaremos el valor del primer linkno que nos encontramos igual a 0 y el segundo igual a 1.
>
> Para el siguiente elemento [i,j] de la matriz, asignaremos los linkno empezando por el valor siguiente al último asignado,

es decir, si antes asignamos 0 y 1, ahora empezaremos asignando el valor 2, y así sucesivamente.

Cada vez que cambiamos de fila en la matriz, y por tanto de junction, se vuelve a empezar asignando el valor 0 al primer atributo linkno que se modifique [40].

2.3 Ejecución del proceso

Para ejecutar el proceso de transformación de la red y obtención de los parámetros necesarios de cara a la implantación del sistema en un nuevo escenario se requiere conectarse mediante una red VPN al servidor utilizado por la Ciudad de Málaga para llevar a cabo el proyecto "SmartCity Málaga", una vez en este, se han llevar a cabo los siguientes pasos [40]:

- Abrir la utilidad de software libre WinSCP [38] y acceder al servidor DESA-SMCT usando el protocolo SFTP.

- Copiar el documento XML que contiene la red previa transformación de sumo al formato *.net.xml al directorio:

  ```
  /home/smartcity/SUMO/Entrada
  ```

- Iniciar la utilidad PuTTY[39]], también de software libre y acceder al servidor DESA-SMCT usando el protocolo SSH.

- Ejecutar el script de transformación de la siguiente manera (incluyendo el punto inicial):

  ```
  . /home/smartcity/SUMO/Scripts/transformar.sh
  ```

- Esperar varios minutos a que acabe y desconectar PuTTY ejecutando el siguiente comando:

  ```
  exit
  ```

- Recoger los resultados con WinSCP desde el directorio `/home/smartcity/SUMO/Salida`, o directamente en el buzón de correo electrónico del destinatario.

Capítulo 6 RESULTADOS

1 Introducción

En este capítulo se presentarán los resultados obtenidos de la simulación de cuatro casos. En todos ellos, se comparan los resultados obtenidos por el sistema de control desarrollado con los resultados obtenidos al gestionar los mismos flujos vehiculares con el sistema de control de lazo abierto que implanta automáticamente un algoritmo incluido en la aplicación del simulador netconvert.exe durante la transformación del mapa en la que este se convierte en apto para realizar sobre él simulaciones con SUMO.

Antes de pasar a analizar los resultados se han de tener en cuenta varios aspectos.

Cuando se indica que en todos los casos ambos sistemas gestionan los mismos flujos vehiculares, lo que se quiere decir es que se gestionan flujos definidos por las mismas funciones y los mismos parámetros estadísticos, así se explican las pequeñas variaciones en número de coches que se observan en los resultados.

En todos los casos excepto en el cuatro se inyectan rutas durante media hora de simulación, tomando valores para los resultados desde el minuto quince, es decir, se tomaron valores a partir de los primeros quince minutos de simulación, y la toma de estos se extendió hasta el momento en el que el sistema de gestión fue capaz de "limpiar la red". Así, y aunque en algunas simulaciones el número de segundos durante el que se prolonga la toma de resultados sea mayor que novecientos, se ha de tener en cuenta que las rutas se dejaron de generar a partir del segundo mil

ochocientos de simulación, o lo que es lo mismo, el segundo novecientos de la toma de resultados, y por tanto, el resto de tiempo incluido en la toma de resultados se debe a que el sistema de gestión lo necesitó para "limpiar" la red, y no a que durante este tiempo se generasen vehículos.

En el caso cuatro se inyectaron rutas durante novecientos segundos, tomando resultados desde el segundo trescientos hasta que el sistema de gestión correspondiente fue capaz de "limpiar" la red.

Los resultados que se han obtenido en los cuatro casos son el número de coches gestionados, el tiempo de toma de resultados, el tiempo medio que un vehículo permanece en la red, el número medio de vehículos en la red y las emisiones y consumos de estos tanto a lo largo del tiempo como en total.

Los flujos base o densidades de tráfico base se han definido mediante el factor de densidad de flujo base, que establece la probabilidad de que sea creada una nueva ruta en cada segundo de simulación según la siguiente ecuación:

$$P[n] = P[n-1] \cdot k^n \; \forall \; n \geq 1, n \in \mathbb{N}$$

Ecuación 49. Probabilidad de creación de rutas en función del parámetro k

En la que P[n] representa la probabilidad de que "n" vehículos sean creados en cualquier segundo de simulación, y k es el factor que se introduce como "densidad de tráfico base" en la interfaz de usuario dividido entre cien.

Así, y al corresponder a los distintos casos unos valores de "densidad de tráfico base" de 35, 40, 5 y 17 respectivamente, se muestran a

continuación en la Tabla 2 las probabilidades de creación de uno o varios nuevos vehículos en cada segundo en los distintos casos.

	Caso 1	*Caso 2*	*Caso 3*	*Caso 4*
Densidad de tráfico base	35	40	5	17
k	0.35	0.4	0.05	0.17
P(1)	0.3500	0.4000	0.0500	0.1700
P(2)	0.0429	0.0640	0.0001	0.0049
P(3)	0.0018	0.0041	0.0000	0.0000
P(4)	0.0000	0.0001	0.0000	0.0000
P(5)	0.0000	0.0000	0.0000	0.0000

Tabla 2. Probabilidades de generación de "n" rutas en un segundo cualquiera para los tres casos simulados

También se ha de tener en cuenta que en los casos 3 y 4 se generó una "ruta especial" con el fin de simular un evento multitudinario para probar en este tipo de casos el sistema de control. Este número de coches no se tiene en cuenta en las probabilidades anteriormente descritas.

Los tres primeros casos han sido simulados sobre el Escenario Parcial, ya que, debido a la fecha de entrega del presente proyecto, no se pudo reservar un tiempo para la obtención de resultados tras la validación del sistema de control en el Barrio Completo. De todas formas, para demostrar la validez del sistema en el Barrio de la Misericordia, el Caso 4 ha sido simulado sobre este escenario.

2 Resultados para el Caso 1

En el Caso 1 se pone a prueba el sistema desarrollado ante un flujo de coches elevado y variado en cuanto a orígenes y destinos, este flujo se pretende comparar con los flujos que se pueden dar en una ciudad en hora punta un día laborable.

Así, para simular este estado del tráfico, se optó por incluir un flujo base del 35%, o lo que es lo mismo, un factor k igual a 0.35. las probabilidades de creación de una o varias nuevas rutas para cada segundo bajo estas condiciones se pueden ver en la Tabla 2.

Sin más dilaciones se pasan a exponer los resultados obtenidos.

En la Tabla 3 se muestra la comparación del número de coches que se crearon durante la simulación, la duración de esta, el número medio de coches en la red, el tiempo medio de permanencia en la red de los coches, el CO_2 emitido durante la toma de resultados y el combustible consumido durante la toma de resultados.

	Sistema de Gestión Inteligente	*Sistema de Gestión de Lazo Abierto*	*Mejoras obtenidas*
Nº de coches	337	348	
Tiempo de simulación	1005	1910	
Tiempo necesario para despejar la red tras el cesar la inserción de rutas	105	1010	89.60%
Nº medio de vehículos en la red	29.78	38.16	21.96%
Tiempo medio de permanencia de los vehículos en la red	88.82	209.42	57.59%
CO2 emitido (g)	54.06	69.15	21.82%
Consumo de combustible (l)	21.17	25.24	16.11%

Tabla 3. Resultados para el Caso 1

En las dos siguientes gráficas se observan tanto las emisiones de CO_2 como los consumos de combustible frente al tiempo. Se puede apreciar claramente como el sistema de gestión inteligente desarrollado es capaz de despejar el tráfico en menos tiempo tras la finalización de la creación de vehículos por parte del simulador.

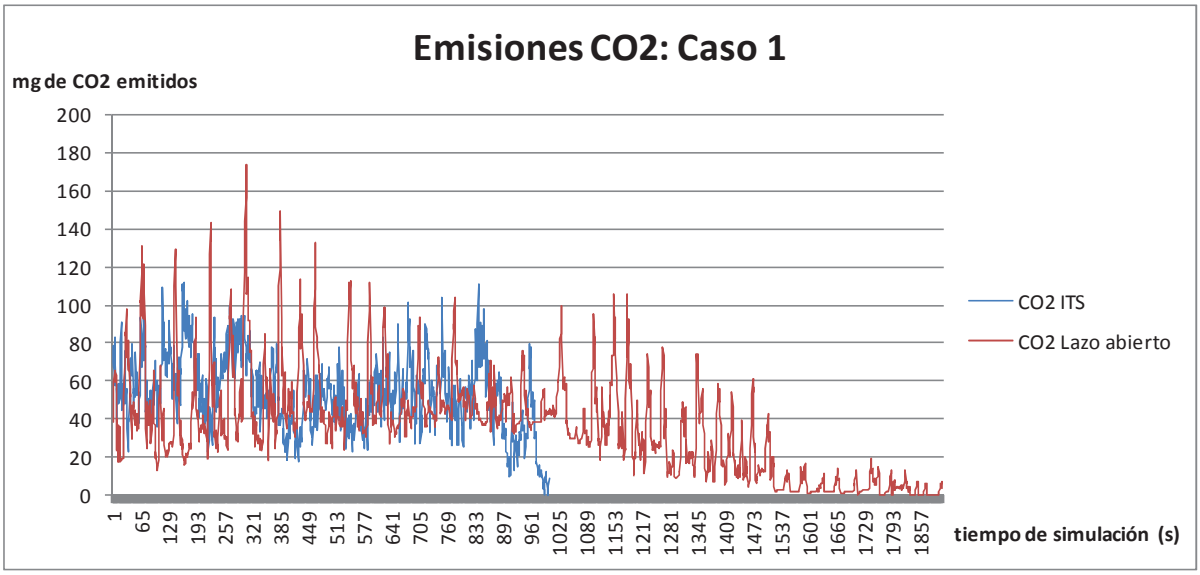


Ilustración 13. Emisiones de CO2 frente al tiempo para el Caso 1

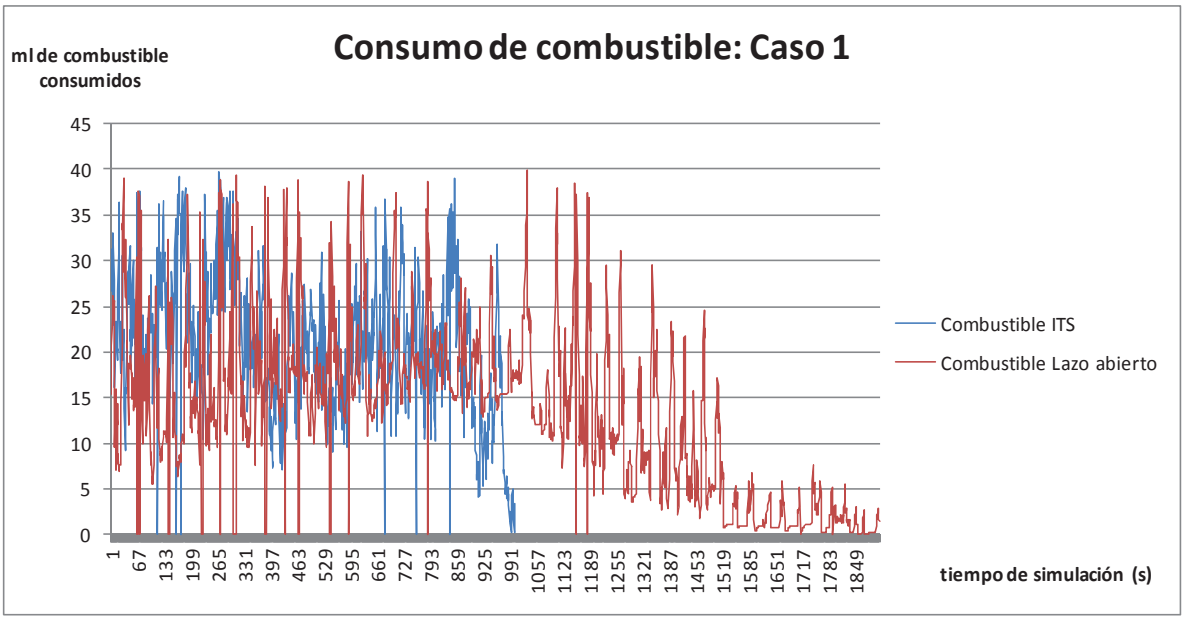


Ilustración 14. Consumo de combustible frente al tiempo para el Caso1

3 Resultados para el Caso 2

En el Caso 2 se puso a prueba el sistema desarrollado ante un flujo vehicular muy elevado y variable en cuanto a orígenes y destinos. El establecimiento de este flujo se llevó a cabo definiendo en la interfaz de usuario un valor para el flujo base del 40%, o lo que es lo mismo, un factor *k* igual a 0.4. Las probabilidades de creación de una o varias nuevas rutas para cada segundo bajo estas condiciones se pueden ver en la Tabla 2.

Este flujo simulado provoca un atasco en un escenario controlado por un sistema de lazo abierto, pero el sistema de control desarrollado es capaz de gestionarlo eficazmente evitando la formación de este.

Al ser capaz de evitar esta formación de atascos, es capaz de reducir enormemente los valores tanto de tiempo necesario para despejar la red como de número medio de vehículos en la red, tiempo medio de permanencia de los vehículos en la red, CO_2 emitido y combustible consumido representando un gran ahorro frente a un sistema de lazo abierto tradicional, como se puede observar en la Tabla 4.

	Sistema de Gestión Inteligente	*Sistema de Gestión de Lazo Abierto*	*Mejoras obtenidas*
Nº de coches	411	415	
Tiempo de simulación	1120	2710	
Tiempo necesario para despejar la red tras el cesar la inserción de rutas	220	1810	87.85%
Nº medio de vehículos en la red	32.05	89.76	64.29%
Tiempo medio de permanencia de los vehículos en la red	87.34	586.16	85.10%
CO2 emitido (g)	65.73	158.97	58.65%
Consumo de combustible (l)	26.23	54.72	52.06%

Tabla 4. Resultados para el Caso 2

Los mismos ahorros se pueden apreciar también en la representación de emisiones de CO_2 y consumos de combustible frente al tiempo, pudiendo observarse en estas gráficas las enormes diferencias en cuanto a tiempo necesario para despejar la red entre ambos sistemas.

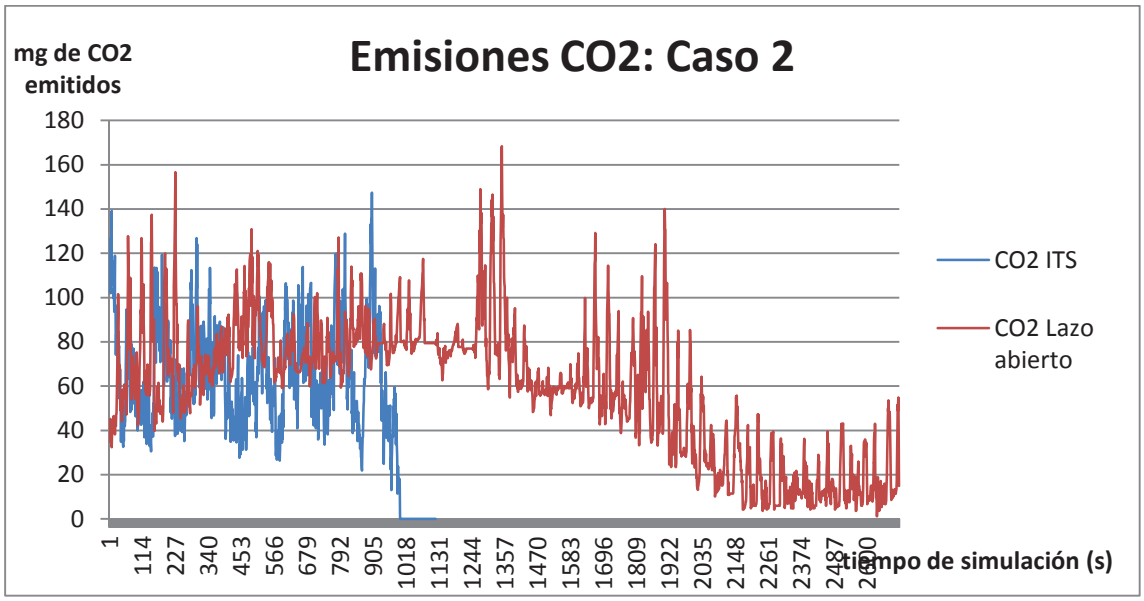


Ilustración 15. Emisiones de CO2 frente al tiempo para el Caso 2

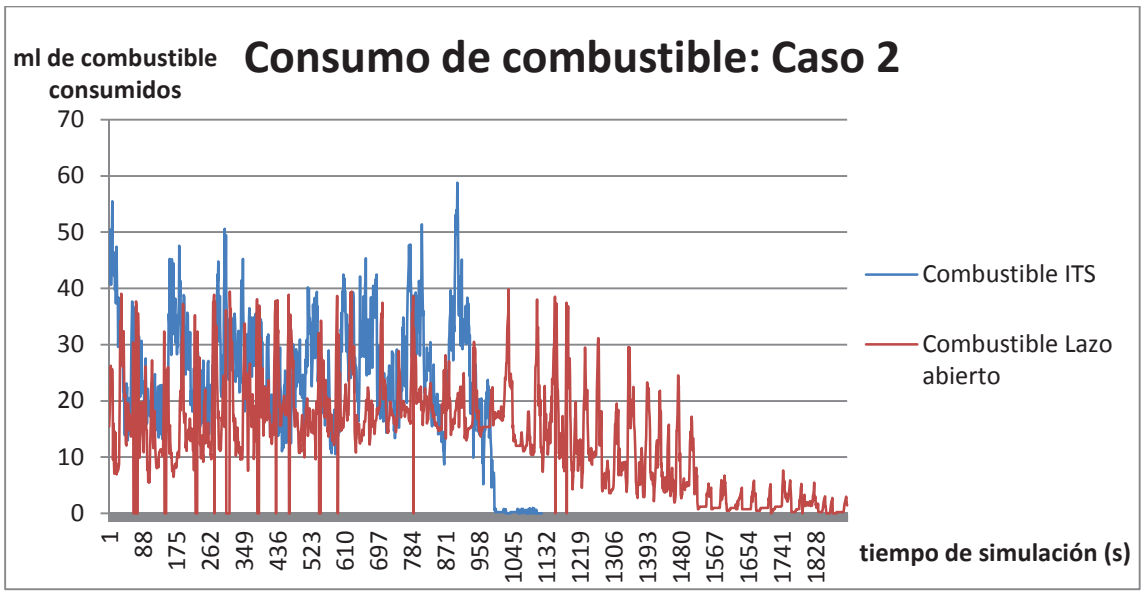


Ilustración 16. Consumo de combustible frente al tiempo para el Caso 2

4 Resultados para el Caso 3

Con la simulación del Caso 3, se pretende simular un evento multitudinario y observar el comportamiento del sistema frente a este tipo de situaciones. Para ello, se estableció un flujo de veinte vehículos por minuto con una ruta concreta, cuyo origen y destino se pueden apreciar en la Ilustración 17. Asimismo se completó la simulación fijando un valor para el flujo base del 5%, o lo que es lo mismo, un factor k igual a 0.05, esto provocó la aparición de un pequeño flujo variable que provocó un aumento de la capacidad de computación necesaria. Las probabilidades de creación de una o varias nuevas rutas para cada segundo bajo estas condiciones de k igual a 0.05 se pueden ver en la Tabla 2.

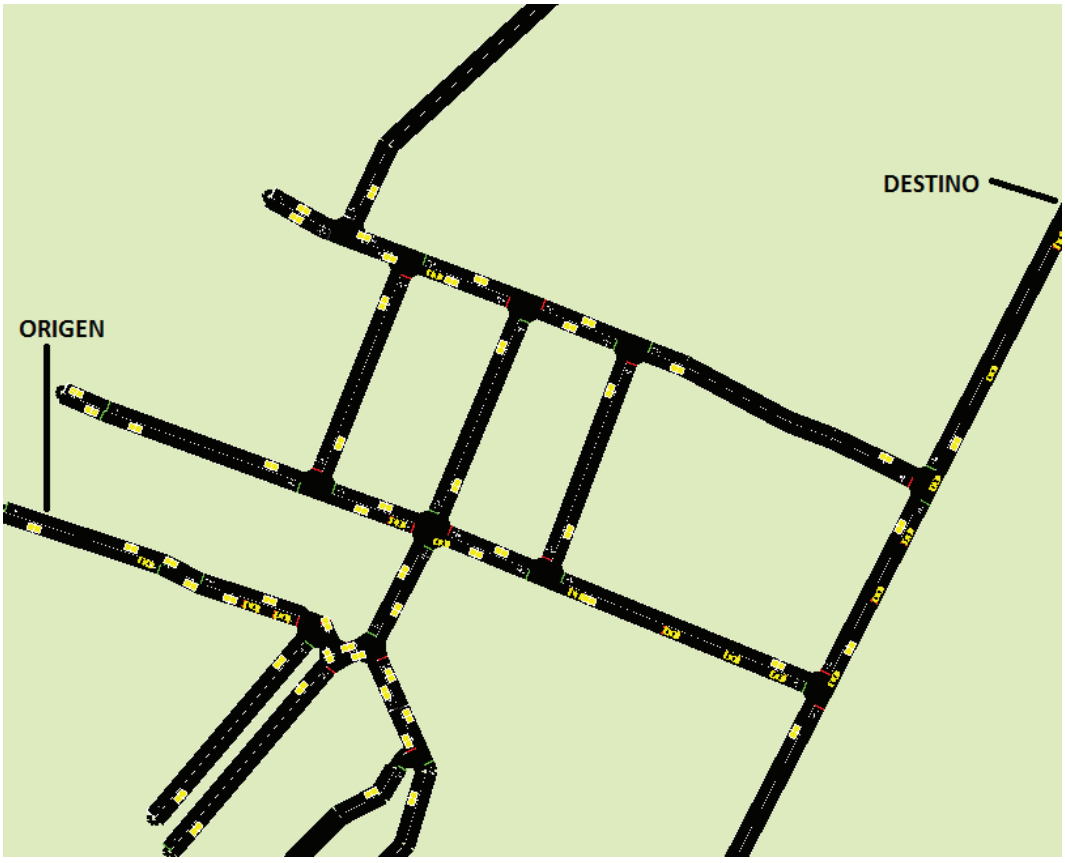


Ilustración 17. Origen y destino de la ruta especial simulada para el Caso 3

Bajo estos flujos vehiculares definidos, ninguno de los sistemas es capaz de gestionar el estado del tráfico de forma que no se formen atascos. Esto no se debe a ninguna deficiencia en el sistema de control desarrollado, sino a que, como se puede observar en el Capítulo 4 de esta Memoria, el sistema de control desarrollado tiene restricciones que limitan el establecimiento de un semáforo en un estado de verde constante, así, el semáforo ha de mantenerse en rojo una determinada fracción de tiempo en cada ciclo de cara a permitir la circulación de los peatones. Es por esto y por la gran cantidad de vehículos que se generaron para una ruta concreta por lo que el sistema de gestión desarrollado presenta unos valores de tiempo necesario para despejar la red, número medio de vehículos en la red, tiempo medio de permanencia de los vehículos, CO_2 emitido y combustible consumido más elevados que en otros casos. Pese al incremento de estos valores respecto a los Casos 1 y 2, el Sistema ha demostrado de todas formas un comportamiento mucho más eficiente que el de un sistema tradicional de lazo abierto, incrementándose también las mejoras porcentuales respecto a los Casos 1 y 2, como se puede ver en la Tabla 5.

	Sistema de Gestión Inteligente	*Sistema de Gestión de Lazo Abierto*	*Mejoras obtenidas*
Nº de coches	308	303	
Tiempo de simulación	3080	19900	
Tiempo necesario para despejar la red tras el cesar la inserción de rutas	2180	19000	88.53%
Nº medio de vehículos en la red	13.54	14.99	9.67%
Tiempo medio de permanencia de los vehículos en la red	135.37	984.62	86.25%
CO2 emitido (g)	83.32	139.84	40.42%
Consumo de combustible (l)	33.22	59.28	43.97%

Tabla 5. Resultados para el Caso 3

Un aspecto interesante que se puede observar durante la simulación son los mayores consumos de combustible y emisiones de CO_2 por unidad de tiempo que tienen lugar cuando los semáforos son gestionados por el sistema optimizado, pese a ser tanto el consumo total como las emisiones totales menores con este sistema.

Esto se debe a que la velocidad media de los vehículos es mayor cuando los semáforos son gestionados por sistema de control inteligente, y mientras que, en la simulación gestionada por el sistema de control de lazo abierto los coches se encuentran parados o circulando a velocidades muy bajas en un atasco, en la simulación en la que el sistema de control inteligente toma el control de los semáforos los vehículos se encuentran circulando a velocidades mayores, por decirlo así, en un atasco "más rápido", es por esto que emiten más CO_2 y consumen más combustible por unidad de tiempo, pero el balance global en ambos casos es menor.

5 Resultados para el Caso 4

Con la simulación del Caso 4 se pretende mostrar el funcionamiento del sistema en un escenario grande y complejo como el Barrio de la Misericordia, en Málaga. Asimismo, los valores de tráfico bajo los cuales es llevada a cabo la simulación pretenden simular un evento multitudinario al tiempo que existe un amplio flujo de vehículos con variabilidad en sus orígenes y destinos. Para ello, se estableció un flujo de quince vehículos por minuto con una ruta concreta, cuyo origen y destino se indican en la Ilustración 18. Se completó la simulación fijando un valor para el flujo base del 17%, o lo que es lo mismo, un factor k igual a 0.17, lo que provocó la aparición de un flujo variable que aumenta los requerimientos de capacidad de computación.

Ilustración 18. Origen y destino de la ruta especial simulada para el Caso 4

Las probabilidades de creación de una o varias nuevas rutas para cada segundo bajo estas condiciones de *k* igual a 0.17 se pueden ver en la Tabla 2.

Estas condiciones de tráfico son probablemente las más exigentes para el simulador, que con una gran cantidad de vehículos recorriendo rutas variables en cuanto a orígenes y destinos ha de ser capaz de discernir la ruta que está siendo recorrida por una gran cantidad de vehículos para poder así asignar a esta un mayor tiempo de verde teniendo en cuenta el resto de flujos vehiculares.

Durante esta simulación, y debido a la elevada cantidad de vehículos que recorren la ruta específica programada, la vía de entrada a la zona a controlar por la que los vehículos que siguen la ruta simulada con una densidad de quince vehículos por minuto entran satura para el Sistema de Gestión de Lazo Abierto, impidiendo así la entrada a nuevos

vehículos hasta que esta quede libre. De esta forma, estos vehículos no son tenidos en cuenta para la simulación al no estar situados en la zona a controlar. Es por esto que el valor del número medio de vehículos en la red es menor para el Sistema de Gestión de Lazo Abierto, y es que, en un caso real, estos vehículos se encontrarían produciendo un atasco en los barrios colindantes.

Esta implicación afecta también al resto de resultados, ya que, desde que el vehículo es generado hasta que consigue entrar en la red pasa un tiempo que no está siendo contabilizado ni para la obtención del tiempo medio de permanencia de los vehículos en la red, ni para la de las emisiones y consumos, …

Pese a esto, pese a la complejidad del escenario y pese a la complejidad de las rutas simuladas, el sistema desarrollado ha demostrado un comportamiento eficaz mejorando, con todo lo anterior, los resultados obtenidos de la simulación controlada con un sistema de gestión de lazo abierto. Así, queda demostrada la validez del sistema.

Los resultados obtenidos se pueden ver en la Tabla6.

	Sistema de Gestión Inteligente	*Sistema de Gestión de Lazo Abierto*	*Mejoras obtenidas*
Nº de coches	225	228	
Tiempo de simulación	1425	3058	53.40%
Tiempo para limpiar la red tras el cese de la creación de rutas	525	2158	75.67%
Nº medio de vehículos en la red	290.46	94.46	-
Tiempo medio de permanencia de los vehículos en la red	1010.65	1293.65	21.88%
CO2 emitido (g)	205.23	263.93	22.24%
Consumo de combustible (l)	81.84	105.22	22.22%

Tabla 6. Resultados para el Caso 4

Capítulo 7 CONCLUSIONES

En este proyecto ha sido desarrollado un sistema de control de tráfico de lazo cerrado, es decir, sensible al estado del tráfico capaz de funcionar en tiempo real en un escenario lo suficientemente grande y complejo como un barrio completo de una ciudad compuesto por 328 cruces, 97 de ellos semaforizados.

Asimismo, se ha conseguido comunicar este sistema con otros sistemas globales de gestión de "Smart Cities", ofreciendo al ciudadano la posibilidad de adaptar la planificación de sus trayectos al estado del tráfico a través del conocimiento de de las colas que se forman en cada semáforo en tiempo real.

La discriminación de escenarios en los que el sistema de control no obtuviese mejoras de cara a que un sistema "tradicional" tomase el control de los semáforos se preparó a través de una variable que se incluyó en el sistema de control y que cambiaría su valor en función del estado del tráfico. Finalmente, el sistema desarrollado es capaz de operar bajo cualquier condición obteniendo mejoras en todos los casos simulados hasta el momento, incluyéndose en estos muy diversas densidades y concentraciones de tráfico.

Podemos concluir entonces que se han cubierto todos los objetivos del proyecto, objetivos opcionales inclusive, que son:

-Optimizar la gestión del tráfico en un escenario que corresponde con un barrio completo de una ciudad.

-Diseñar una arquitectura que permita su implementación sensible al estado del tráfico.

-Integrar dicho sistemas en sistemas globales de gestión de "Smart Cities". (Opcional)

-Discriminar en la optimización escenarios de baja densidad de tráfico de los de alta densidad. (Opcional)

Si bien la discriminación mencionada en el último objetivo no se encuentra implementada en el sistema de control, esto se debe a que finalmente se comprobó que no es necesaria tal discriminación.

También se consiguió en este proyecto un gran avance de cara a la utilización de este sistema, y que se refiere a la automatización de la creación de escenarios para la simulación y la implantación del sistema de control desarrollado en estos como se indicó en el Capítulo 5 de esta memoria.

Así, se podrían de destacar las siguientes como conclusiones finales:

-Se ha desarrollado un sistema de control inteligente, sensible al estado del tráfico, capaz de operar en una red grande y compleja como el Barrio de la Misericordia de Málaga.

-El sistema extrae información acerca del estado del tráfico para poder mostrársela a los ciudadanos y que estos puedan utilizar esta información para planificar sus rutas.

-Se ha desarrollado un método a través del cual el sistema se puede implementar de forma sencilla y rápida en un nuevo escenario.

-El sistema reduce el número medio de vehículos en la red entre un 9% y un 65%.

-El sistema reduce el tiempo invertido en los desplazamientos entre un 57% y un 87%.

-El sistema reduce las emisiones de CO2 entre un 21% y un 41%.

-El sistema reduce el consumo de combustible entre un 16 y un 53%.

Capítulo 8 FUTUROS DESARROLLOS

En este apartado se tratarán de describir las posibles mejoras a realizar sobre este proyecto que, por no estar incluidas en los objetivos y por la fecha de entrega del proyecto, no se han podido llevar a cabo.

1 Simulador

Durante el desarrollo del proyecto se ha podido comprobar la validez en cuanto a funcionalidad y rapidez del sistema de control desarrollado, asimismo, se ha percibido la lentitud de la simulación debido a la ingente cantidad de datos transmitidos entre este y el sistema de control vía TraCI.

Se propone como futuro desarrollo, para poder realizar pruebas y simulaciones sobre escenarios mayores en tamaño al utilizado para este proyecto, intentar utilizar las comunicaciones vía TraCI lo mínimo posible, ya que, ante un escenario con una cantidad de sensores más elevada que la de los escenarios simulados en este proyecto, se prevé que la velocidad de simulación disminuya alargando la duración del proceso de toma de resultados pese al correcto funcionamiento del sistema desarrollado.

Se propone para evitar esta carga de comunicaciones mediante TraCI utilizar un lector de archivos XML para leer el archivo "detectores.out.xml" en el que el simulador incluye todas las lecturas de los detectores automáticamente durante la simulación.

Una vez este haya sido desarrollado, se podrán validar con mayor rapidez también los parámetros extraídos para cada escenario y las transformaciones realizadas en este a través de simulaciones más rápidas.

2 Capacidad de las vías

Se ha comprobado que, cuando se forman atascos en cualquiera de los dos sistemas, tanto en el desarrollado en este proyecto como en el de lazo abierto utilizado para analizar las mejoras obtenidas, la causa de estos es principalmente una cola que excede la capacidad de la vía, es decir una cola que "rebosa" la calle que ocupa interfiriendo en el tráfico en los cruces adyacentes, así, y pese a la baja tasa de atascos presentada por el sistema de control desarrollado en comparación con los sistemas de lazo abierto, se propone incluir en la función objetivo del modelo matemático desarrollado una penalización para todas aquellas colas que puedan exceder la capacidad de la vía.

La capacidad de la vía se puede obtener fácilmente. De la definición de las calles expuesta en el apartado 3 del Capítulo 5 de esta memoria se puede extraer la longitud de cada calle (campo "length"). Una vez dividamos esta entre la longitud media de los vehículos más un pequeño espacio que separa a cada vehículo de su adyacente tendremos la capacidad de la vía medida en vehículos. Una vez se tenga este valor, solo se ha de comprobar si la cola supera o no esta capacidad, y penalizar la función objetivo en caso de una respuesta afirmativa.

BIBLIOGRAFÍA

[1] A. Saiz Peguero, "SMARTCITY: Sistema de gestión de tráfico inteligente."

[2] D. Robles, P. Nanez, and N. Quijano. "Urban Traffic Control and Simulation in Colombia: Literature Review". rev.ing., Jan./June 2009, no.29, p.59-69. ISSN 0121-4993.

[3] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and Y. Wang, "Review of road traffic control strategies," Proc. IEEE, vol. 91, no. 12, pp. 2043–2067, 2003.

[4] Al-Mudhaffar. "Impacts of traffic signal control strategies". Ph.D. dissertation, Royal Institute of Technology, 2006.

[5] K. Aboudolas, M. Papageorgiou and E. Kosmatopoulos. "Control and optimization methods for traffic signal control in large-scale congested urban road networks". Proceedings of the 2007 American Control Conference, July 2007, pp. 3132–3138.

[6] S. Algers, E. Bernauer, M. Boero, L. Breheret, C. Di Taranto, M. Dougherty, K. Fox and J. Gabard. "Review of micro-simulation models". Rep. SMARTEST/D, Vol. 3, 1998, pp. 1–111.

[7] R.B. Allsop, "SIGSET: A computer program for calculating traffic capacity of signal-controlled road junctions." Traffic Eng. Control, vol. 12 pp. 58-60, 1971.

[8] R.B. Allsop, "SIGCAP: A computer program for assessing the traffic capacity of signal-controlled road junctions." Traffic Eng. Control, vol. 17, pp. 338-341, 1976.

[9] J.D.C. Little, M.D. Kelson and N.H. Gartner, "MAXBAND: A program for setting signals on arteries and triangular networks", Transport. Res. Rec., no. 795, pp40-46, 1981.

[10] "UTCS Technical Specifications"

[11] K. Courage and C.E. Wallace, "TRANSYT-7F User´s Guide, Federal Highway Administration", Washington D.C., 1991.

[12] P. T. Martin, J. Perrin, B. Rama Chilukuri, C. Jhaveri and Y. Feng. "Adaptive Signal Control II". University of Utah Traffic Lab, Department of Civil and Environmental Engineering, Salt Lake City, Utah, January 2003.

[13] M. Dotoli, M. Fanti and C. Meloni. "A signal timing plan formulation for urban traffic control". Control Engineering Practice. Vol. 14, No. 11, 2006, pp. 1297–1311.

[14] S. Adams and L. Yu. "An evaluation of traffic simulation models for supporting its development". Center for Transportation Training and Research, Texas Southern University, Tech. Rep., October 2000.

[15] F. Wang. "Agent-based control for networked traffic management systems". IEEE Intelligent Systems, Vol. 20, No. 5, 2005, pp. 92–96.

[16] A. Di Febbraro and N. Sacco. "Closed Loop Control of Special Classes of Vehicles". Proceedings of the 8th International IEEE Conference on Intelligent Transportation Systems, 2005.

[17] J.R. Peirce and P.J. Webb. "MOVA control of isolated traffic signals recent experience". Third International Conference on Road Traffic Control pp. 110-113, 1990.

[18] Página principal de SCOOT: "http://www.scoot-utc.com"

[19] N. B. Bounsell[1], F. N. McLedo[1] and P. Burton[2]. "SCOOT: A traffic database" .[1]University of Southampton, UK.[2] Transport and Road Research Laboratory, UK.

[20] P.R. Lowrie. "SCATS: The Sydney co-ordinated adaptive traffic system. Principles, methodology, algorithms" in Proc. IIEE International Conference on Road Traffic Signalling, London, UK, pp. 67-70, 1982.

[21] P. Mirchandani and L. Head, "RHODES: A real-time traffic signal control system: Architecture, algorithms and analysis." TRISTAN III, Triennial Symp. Transport Analisis, vol. 2, San Juan, Puerto Rico, 1998.

[22] F. Busch and G. Kruse. "MOTION for SITRAFFIC A modern approach to urban traffic control". Proceedings of the IEEE Conference on Intelligent Transportation Systems, California, pp. 61-64, 2001.

[23] V. Dinopoulou, D. Diakaki and M. Papageorgiou. "Applications of the urban traffic control strategy TUC." European Journal of Operational Research, vol. 175, issue 3, pp. 1652-1665, 2006.

[24] N.H. Gartner, "OPAC: A demand-responsive strategy for traffic signal control", Transport Res. Rec., nº 906, pp. 75-84, 1983.

[25] D. Krajzewicz, G. Hertkorn, C. Rössel and P. Wagner. "SUMO (Simulation of Urban MObility); An open-source traffic simulation". SCS European Publishing House, pp. 183-187, 4th Middle East Symposium on Simulation and Modelling, Sept. 2002, Sharjah / United Arab Emirates.

[26] L. Dimitriou, T. Tsekeris and A. Stathopoulos. "Genetic-algorithm-based micro-simulation approach for estimating turning proportions at signalized intersections". Control in Transportation Systems, Volume 11, Part 1.

[27] D. Krajzewicz, M. Behrisch. "SUMO – Simulation of Urban Mobility – User Documentation".

[28] Página principal de SUMO. "http://sumo.sourceforge.net/"

[29] A. Wegener, M. Piorkowski, M. Raya, H. Hellbrüc, S. Fischer and J. P. Hubaux. "TraCI: An Interface for Coupling Road Traffic and Network Simulators". Proceedings of the 11th communications and networking simulation symposium, April 14-17, 2008, Ottawa, Canada.

[30] R. González "Python para todos"

[31] G. van Rossum. "The Python Language Reference. Release 2.7.1". Python Software Foundation.

[32] M. Pilgrim. "Dive into Python". Apress, May 2004 Edition

[33] J. Walkenbach. "Excel 2007 VBA Programming for Dummies". Wiley Publishing, Inc.

[34] [46] IBM ILOG. "IBM ILOG OPL Getting Started with the OPL IDE". IBM ILOG OPLV6.3. 2009.

[35] IBM ILOG. "IBM ILOG OPL Language Reference Manual". IBM ILOG OPLV6.3. 2009.

[36] IBM ILOG. "IBM ILOG OPL IDE Reference". IBM ILOG OPLV6.3. 2009

[37] Página principal de la herramienta IBM ILOG CPLEX Optimization Studio: "http://www-01.ibm.com/software/integration/optimization/cplex-optimization-studio/".

[38] Página principal de WinSCP. "http://winscp.net/".

[39] Página principal de PuTTY "http://putty.org/".

[40] A. García, C. Seoane "Documentación del proceso de transformación para simulador de tráfico con IBM InfoSphere DataStage"

Parte II MANUAL DE USUARIO

Capítulo 1 MANUAL DE USUARIO

1 Introducción

Para poder ejecutar este proyecto y realizar desarrollos o pruebas sobre él se han de tener instalados los siguientes programas:

-SUMO (Simulation of Urban Mobility) versión 0.12.2 o superior.

-Python, versión 2.7 o superior.

-Notepad o programa similar que permita manejar documentos con extensión *.txt

-IBM ILOG CLEPX Optimization Studio versión 12.2 o superior

-Microsoft Office Excel 2007 o superior, en el que se incluye VBA

-WinSCP versión 4.3.7 o superior

-PuTTY versión 0.62 o superior

2 Implementación del sistema en un nuevo escenario

Para la implementación del sistema en un nuevo escenario, se podrán utilizar los mismos sistema de control, interfaz de usuario y modelo matemático que se han desarrollado a lo largo de este proyecto.

Para la adaptación de estos a un nuevo escenario se han de realizar los siguientes pasos:

La primera acción a realizar consiste en extraer el mapa de la zona a controlar en formato XML del proveedor de mapas online OpenStreetMap. Lo primero que se ha de hacer antes de exportarlo es comprobar la semaforización de la zona. En caso de los semáforos no se

encuentren incluidos en el mapa debemos de añadirlos nosotros en el propio OpenStreetMap, tratando de reproducir con la mayor fidelidad posible la colocación de semáforos en el área en la que vayamos a llevar a cabo nuestra simulación.

En caso de que fuese necesaria la semaforización de la zona, esta se realiza como usuario de OpenStreetMap, por lo que hemos de registrarnos. Tras esto, se accede con el nombre de usuario y contraseña, y se selecciona en la barra superior la opción editar, que solo estará disponible si hemos hecho el zoom suficiente sobre la zona a semaforizar. En el menú desplegable se escoge la opción "Editar con Potlatch 2 (editor en el navegador)". Una vez elegido, se seleccionan los cruces que se quiera semaforizar uno por uno, y en la pestaña "Basic" en el menú que aparecerá a la izquierda de la pantalla se establece el valor del campo "Traffic Signals" como "YES". Una vez hallamos semaforizado la zona basta con guardar los cambios pulsando el botón "Guardar" en la parte superior izquierda de la pantalla.

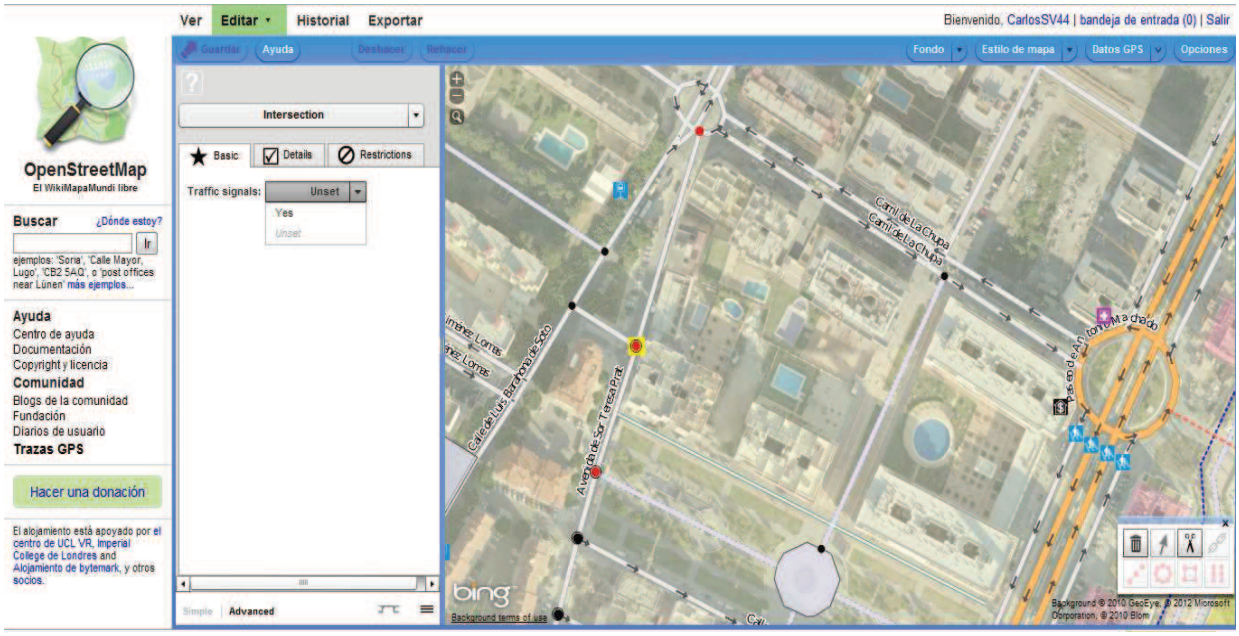


Ilustración 19. Detalle del proceso de colocación de semáforos en OSM

Una vez este la zona ha sido semaforizada si no lo estaba previamente, se exportará en formato *.xml, para ello no es necesario ser usuario de OpenStreetMap, y basta con acceder a la página principal, buscar la zona a semaforizar en el mapa, hacer zoom sobre ella y seleccionar en la barra superior el menú "Exportar". Una vez seleccionado, se marca en el menú que aparecerá en la parte izquierda de la pantalla la opción "Datos en formato OpenStreetMap XML", y se pulsa sobre "Seleccionar a mano otra área en este mismo menú". Ya solo falta arrastrar la selección sobre la parte del mapa que queramos extraer para implantar en ella nuestro sistema de control.

Una vez el mapa ha sido extraído, se le aplicarán las transformaciones necesarias para convertirlo en un mapa cuyo formato acepte SUMO para realizar la simulación en él. Para esto utilizaremos la aplicación netconvert.exe que se incluye en SUMO, y transformaremos el archivo escribiendo lo siguiente en la línea de comandos:

```
netconvert --osm-files map osm.xml –o map.net.xml
```

sustituyendo "map" por el nombre del fichero que queremos transformar.

Otra opción para la realización de esta última transformación consiste en la inclusión del mapa extraído de OpenStreetMap en la carpeta "Traductor OpenStreetMap", tras esto, la ejecución del archivo por lotes de Windows "CrearMapaSUMO.bat" (creado en este proyecto a tal efecto) creará un nuevo documento con el nombre "misericordia.net.xml", fichero ya apto para realizar sobre él simulaciones.

Una vez tenemos un mapa sobre el que se pueden realizar simulaciones es necesario configurar el sistema de control para el funcionamiento de este en el nuevo escenario.

Lo primero que debemos de hacer es extraer los datos necesarios del mapa y realizar las modificaciones pertinentes en este como se indicó en el Capítulo 5 de la Parte I de la presente memoria.

Así, una vez tengamos el archivo de la red en formato *.net.xml deben de realizarse las siguientes tareas:

-Conectarse mediante una conexión VPN al servidor utilizado por la Ciudad de Málaga para llevar a cabo el proyecto "SmartCity Málaga".

-Copiar mediante la utilidad de software libre WinSCP el documento XML que contiene la red previa transformación de sumo al formato *.net.xml al directorio:

```
/home/smartcity/SUMO/Entrada
```

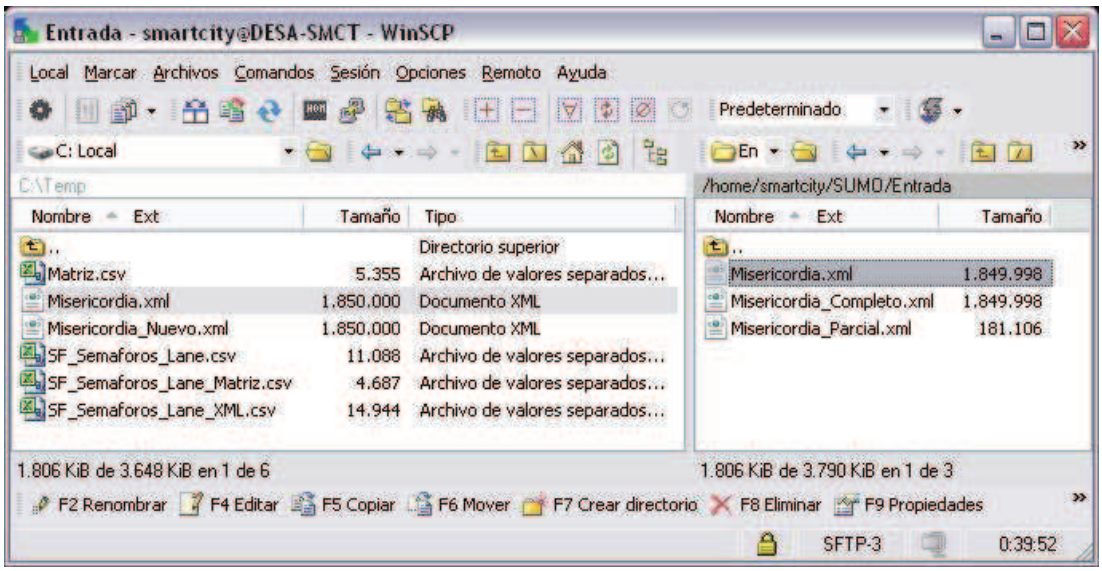

Entrada - smartcity@DESA-SMCT - WinSCP
Local Marcar Archivos Comandos Sesión Opciones Remoto Ayuda
Predeterminado
C: Local
En
C:\Temp
/home/smartcity/SUMO/Entrada
Nombre Ext Tamaño Tipo
Directorio superior
Matriz.csv 5.355 Archivo de valores separados...
Misericordia.xml 1.850.000 Documento XML
Misericordia_Nuevo.xml 1.850.000 Documento XML
SF_Semaforos_Lane.csv 11.088 Archivo de valores separados...
SF_Semaforos_Lane_Matriz.csv 4.687 Archivo de valores separados...
SF_Semaforos_Lane_XML.csv 14.944 Archivo de valores separados...
Nombre Ext Tamaño
Misericordia.xml 1.849.998
Misericordia_Completo.xml 1.849.998
Misericordia_Parcial.xml 181.106
1.806 KiB de 3.648 KiB en 1 de 6
1.806 KiB de 3.790 KiB en 1 de 3
F2 Renombrar F4 Editar F5 Copiar F6 Mover F7 Crear directorio F8 Eliminar F9 Propiedades
SFTP-3
0:39:52

Ilustración 20. Detalle de la utilización de WinSCP

-Acceder al servidor Unix mediante la utilidad PuTTY [39], también de software libre.

-Ejecutar el script de transformación de la siguiente manera (incluyendo el punto inicial):

```
. /home/smartcity/SUMO/Scripts/transformar.sh
```

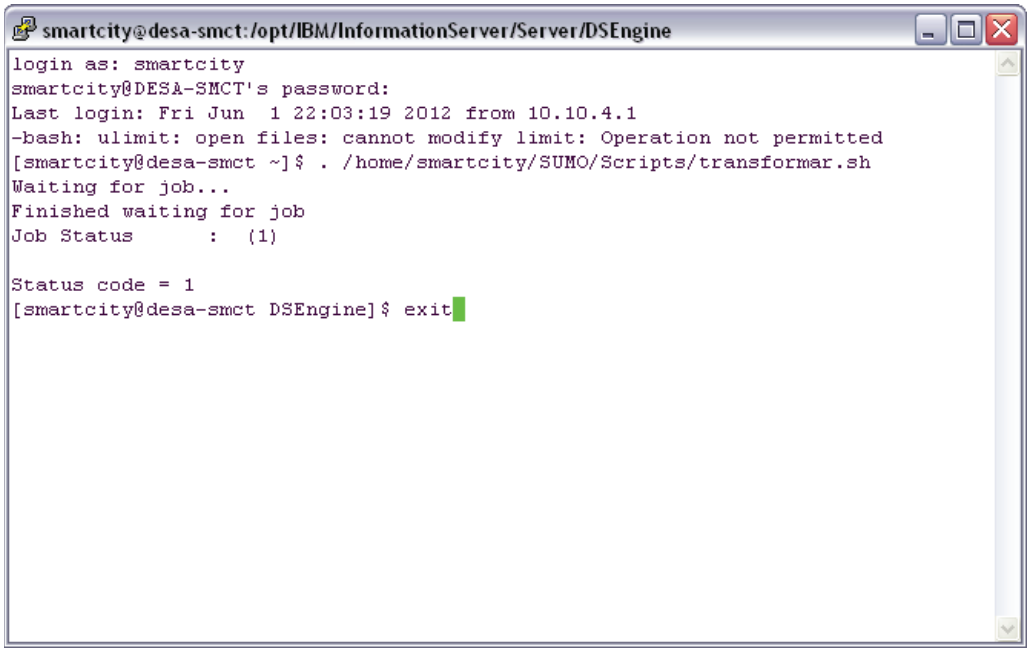

smartcity@desa-smct:/opt/IBM/InformationServer/Server/DSEngine
login as: smartcity
smartcity@DESA-SMCT's password:
Last login: Fri Jun 1 22:03:19 2012 from 10.10.4.1
-bash: ulimit: open files: cannot modify limit: Operation not permitted
[smartcity@desa-smct ~]$. /home/smartcity/SUMO/Scripts/transformar.sh
Waiting for job...
Finished waiting for job
Job Status : (1)
Status code = 1
[smartcity@desa-smct DSEngine]$ exit

Ilustración 21. Detalle de la utilización de PuTTY

-Esperar varios minutos a que acabe y desconectar PuTTY ejecutando el siguiente comando:

```
exit
```

-Recoger los resultados con WinSCP desde el directorio `/home/smartcity/SUMO/Salida` o directamente en el buzón de correo electrónico del destinatario.

Una vez hayamos realizado estas operaciones, tendremos a nuestra disposición los siguientes ficheros:

-Misericordia_nuevo.xml

-SF_EdgeLane_Contorno.csv

-SF_EdgeLane_Estandar.csv

-SF_EdgeLane_Vectores.csv

-SF_Edges_Contorno_1.csv

-SF_Edges_Contorno_2.csv

-SF_Edges_Direccion_1.csv

-SF_Edges_Direccion_2.csv

-SF_Edges_Predecesores.csv

-SF_Edges_Total.csv

-SF_Junctions_Edges.csv

-SF_Junctions_Sin_Semaforizar.csv

-SF_Junctions_Total.csv

-SF_Semaforos_Lane.csv

-SF_Semaforos_Lane.xml

-SF_Semaforos_Lane_Matriz.csv

-Matriz.csv

Mediante el uso de estos y el sistema ya desarrollado para el Barrio de la Misericordia se podrá realizar la implantación del sistema en un nuevo escenario realizando las modificaciones que se describen a continuación:

Cambiar el nombre a "Misericordia_nuevo.xml", e incluirlo en la carpeta en la que está contenido el sistema de control bajo el nombre "Misericordia.net.xml" reemplazando al archivo existente.

Crear un archivo que tenga como contenido los archivos "SF_EdgeLane_Contorno.csv" y "SF_EdgeLane_Estandar.csv" (copiando el contenido de un archivo a continuación del otro). Se tendrá que realizar en este archivo combinado el remplazo de "`<e1detector`" por "`<e1-detector`". Una vez se tenga el archivo combinado con el reemplazo ya efectuado, se ha de reemplazar el contenido del archivo "Misericordia.det.xml" por el del archivo recién creado por nosotros. Con esta operación, habremos instalado correctamente los sensores en nuestra zona a controlar.

Abrir a continuación el archivo "ControlTrafico.py" para editarlo y realizar las siguientes operaciones:

-Cambiar los valores y longitud del vector "sensores" de forma que se incluyan en el todos los valores de los distintos campos "`id`" que aparecen en el archivo creado previamente por el usuario "Misericordia.det.xml".

-Asignar este mismo vector al definido como "Nombre_sens".

-Sustituir en este vector todos sus valores por cero para obtener el vector que reemplazará a "sumasensor". Este mismo vector de ceros se asignará también al definido como "ocupacsensor".

-Abrir el fichero "SF_Junctions_Total.csv", en el que aparece un número entero, a continuación reemplazar el vector cruces por uno cuyas posiciones sean números enteros consecutivamente hasta alcanzar el valor indicado en "SF_Junctions_Total.csv". Este mismo vector se asignará al definido como "CrucesId".

-Reemplazar en el vector definido anteriormente todos sus valores por ceros, y asignar este nuevo vector al definido como "sumacruce".

-Abrir el archivo "SF_Junctions_Edges" y se crear un vector en el que cada posición se corresponderá con un "id" de un cruce que se encuentra en el archivo antes mencionado. Esta asignación se debe de hacer asegurándose que las posiciones del vector mantienen el mismo orden que el que utiliza el archivo indicado. Una vez hecho esto, se asignará este vector al definido como "NombreCruces".

-Abrir el archivo "SF_Semaforos_Lane.xml", y crear un vector en el que aparezcan por el mismo orden que en el archivo los penúltimos campos de cada línea descrita en el archivo. Este vector se asignará al definido como "NombreCrucesSemaforizados".

- Abrir el archivo "SF_Edges_Total.csv", y se buscar la línea con un valor del segundo campo más alto, tomar ese valor y crear un vector en el que el valor de cada posición sea igual al índice de esta, es decir, la posición cero tendrá valor cero, la uno, uno,… el archivo tendrá una longitud tal que su último elemento sea igual al valor tomado anteriormente del archivo "SF_Edges_Total.csv".

-Abrir el archivo "SF_EdgeLane_Contorno.csv" y crear un vector en el que aparezcan en todas sus posiciones los distintos "id" descritos en el archivo. Una vez creado, asignar este vector al definido como "sensorescontorno".

-Sustituir en el archivo creado previamente todos sus valores por cero y asignarlo a los definidos como "sumasensorcontorno" y "ocupacsensorcontorno".

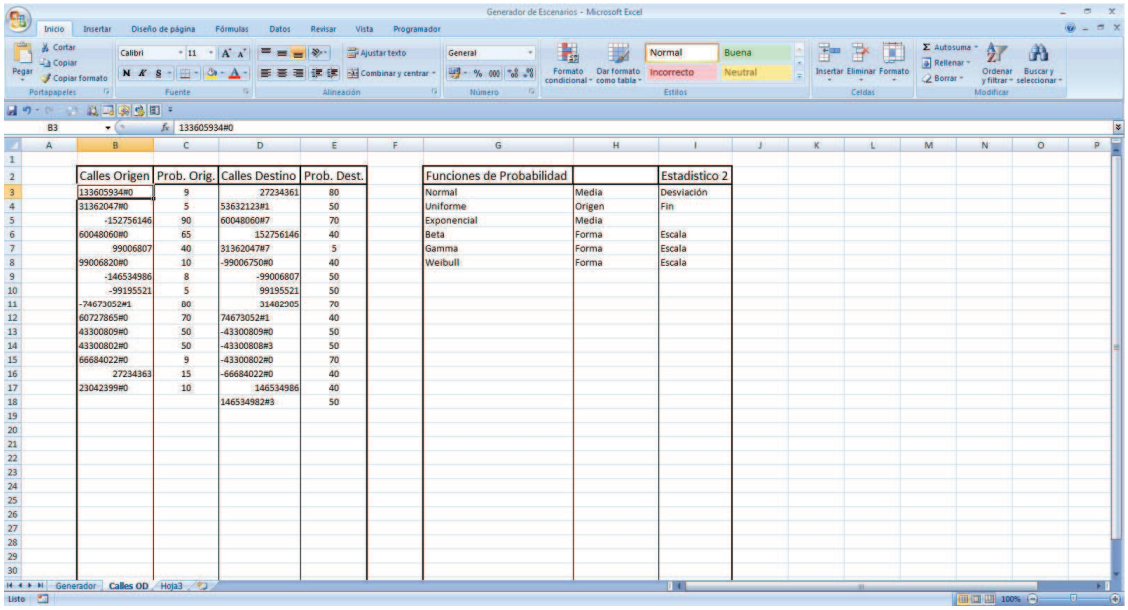
-Por último, crear una matriz con tantas filas como posiciones tiene el vector "cruces" creado previamente y tantas columnas como el vector "direcciones" descrito también anteriormente. Asignar a todas sus posiciones el valor cero y asignar esta matriz a las definidas como "sumaratio", "ratiofinal" y "sumaratio".

-Con estas asignaciones, habremos definido ya todos los vectores necesarios para adaptar nuestro programa "ControlTráfico.py" al nuevo escenario, ahora solamente necesitamos, definir las asignaciones también tras la línea:

```
if (CPLEX_Just_End):
```

En esta parte no es necesario volver a definir todos los vectores, bastará con redefinir "sumacruce", "sumasensor", "ocupacsensor", "ratiofinal", "sumasensorcontorno", "ocupacsensorcontorno" y "sumaratio".

El penúltimo archivo que hemos de modificar es la hoja de cálculo habilitada para macros "Generador de Escenarios", en este archivo, la modificación es muy simple, solamente hemos de ir a la hoja "CalleOD", y modificar en esta los valores de las columnas "Calles Origen" y "Calles Destino" por los valores que ocupan los campos "lane" en el archivo "SF_EdgeLane_Contorno" tras eliminar de estos la parte que sigue a "_" incluyendo este carácter en la eliminación.



Calles Origen	Prob. Orig.	Calles Destino	Prob. Dest.
13360593#0	9	27234361	80
31362047#0	5	53632123#1	50
-152756146	90	60048060#7	70
60048060#0	65	152756146	40
99006807	40	31362047#7	5
99006820#0	10	-99006750#0	40
-146534986	8	-99006807	50
-99195521	5	99195521	50
-74673052#1	80	31402905	70
60727865#0	70	74673052#1	40
43300809#0	50	-43300809#0	50
43300802#0	50	-43300808#3	50
66684022#0	9	-43300802#0	70
27234363	15	-66684022#0	40
23042399#0	10	146534986	40
		146534982#3	50

Funciones de Probabilidad		Estadístico 2
Normal	Media	Desviación
Uniforme	Origen	Fin
Exponencial	Media	
Beta	Forma	Escala
Gamma	Forma	Escala
Weibull	Forma	Escala

Ilustración 22. Detalle de la modificación de la hoja de Excel

Se recomienda la elección de que campos incluir, cuáles no, y con qué probabilidades de origen y destino tras abrir la red "Misericordia.net" con SUMO y comprobar la situación de estas calles haciendo clic con el botón derecho del ratón en las calles que conforman el contorno del escenario. Así podremos diseñar una situación del tráfico más real.

El último archivo que es necesario modificar es el modelo matemático, para ello, abrimos el archivo "ModeloCPLEXBarrioCompleto.dat" o "ModeloCPLEXParcial.dat" según el Sistema que se haya tomado como base para configurar el nuevo.

En este archivo, se han de modificar los siguientes campos:

-Se ha de modificar el valor del vector "Cruces" de forma que este coincida con el vector "cruces" que se ha definido previamente para el archivo "ControlTrafico.py".

-Se ha de modificar el valor del vector "NbCarriles" para que en todas las posiciones de este vector aparezca el valor

correspondiente al número más alto del vector "direcciones" que se ha definido previamente para el archivo "ControlTrafico.py" más uno.

-Se han de modificar los valores pertenecientes al subconjunto "Contorno", haciendo que se incluyan en este los índices de todos aquellos cruces que forman parte del contorno del escenario en el que se pretende implantar el sistema. Para ello, se tomarán los valores a incluir en este parámetro del último campo de cada elemento descrito tanto en el archivo SF_Edges_Contorno_1.csv como en el archivo SF_Edges_Contorno_2.csv.

-Se modificarán los valores del subconjunto "Predecesores", incluyendo dentro de este todos los índices correspondientes al último campo de todos los elementos descritos en el fichero "SF_Edges_Predecesores.csv".

-Para modificar el subconjunto "CarrilesConPredec", se tomará el subconjunto "Predecesores" modificado previamente, y si este está compuesto por combinaciones de dos índices, se tomará el primer índice de cada una de estas combinaciones para incluirlo en "CarrilesConPredec".

-El subconjunto "Carriles1" estará formado por todos los índices que aparecen en el último campo de los elementos descritos en el archivo "SF_Edges_Direccion_1.csv".

-La matriz "NbSemaforos" se corresponderá con la que se incluye en el archivo "Matriz.csv", completando esta con filas de ceros hasta que tenga tantas filas como posiciones distintas tiene el vector "cruces" que se ha definido previamente para el archivo "ControlTrafico.py".

-El vector "NbSemaforosCruce" se modificará para que sus valores se correspondan con las sumas de valores por filas de la matriz "NbSemaforos".

Una vez se hayan realizado todos los cambios indicados, se tendrá el Sistema de Control configurado para funcionar en el nuevo escenario.

Pese a esto, se conveniente realizar varias simulaciones para validar el sistema, ya que en estos archivos puede haber algún pequeño error de construcción. Durante estas simulaciones para validar el sistema, utilizaremos configuraciones de tráfico muy sencillas y observaremos que el Sistema responde como se espera, en caso de que no sea así, haremos uso de los archivos y la explicación del contenido de estos dada en el Capítulo 5 así como en la primera parte del presente capítulo para corregir aquellos pequeños errores que se puedan ir encontrando.

Para tomar resultados o realizar pruebas en este nuevo escenario, se han de seguir las instrucciones indicadas en el siguiente apartado.

3 Realización de pruebas y toma de resultados

Para la realización de pruebas y toma de resultados con nuestro Sistema de Control, se debe escoger apropiadamente la versión que toma resultados o la que no lo hace. La diferencia entre ambas se puede apreciar en el nombre de la carpeta en que están contenidas, y no se recomienda utilizar la versión realizada para tomar resultados en caso de que estos no se quieran obtener por la ralentización que esta toma de resultados produce en la simulación.

Una vez elegido el tipo de simulación que queremos realizar (una prueba del sistema o una toma de resultados), se debe ejecutar el

programa "Generador de Escenarios.xlsm", el cual al abrirse mostrará una ventana informando de que se ha accedido al "Editor de Escenarios". Tras pulsar "Aceptar" y antes de poder generar los escenarios o ejecutar alguna simulación es importante comentar que existen dos hojas de cálculo, una con la interfaz en sí misma ("Generador") y otra con la base de datos del escenario fijo sobre el que crear las rutas ("Calles OD"). En la primera de ellas no debe escribirse en ningún caso, ya que todos los datos se introducen a través de las ventanas que aparecen al pulsar los botones correspondientes. Sin embargo la base de datos si permite su modificación, como se expuso en el apartado 2 del presente capítulo.

Lo mínimo que se debe definir para poder generar el escenario de simulación es el flujo base y la extensión temporal de la simulación. El primero se introducirá utilizando el botón "Definir Flujo Base", el cual al pulsarse mostrará una ventana con un scroll horizontal que permitirá fijar el porcentaje indicado tras pulsar "Aceptar". El tiempo de la simulación se introduce pulsando el botón "Tiempo de Simulación". Al aparecer la ventana se verá que dispone de una casilla de opción que permitirá definir el tiempo mínimo de cambio (1 minuto o 10 minutos) con los scrolls horizontales que también pueden verse y que cambian el tiempo de inicio y fin de la simulación. Se ha programado que la diferencia de estos tiempos no pueda ser menor a 5 minutos, y permite simular el cambio de día (pasar por las 00:00) si el tiempo de inicio es mayor que el de fin. Al pulsar "Aceptar" la información será guardada en el cuadro correspondiente de la primera hoja de la interfaz.

Por otro lado se permite añadir flujos de rutas individuales entre dos puntos del escenario y dos instantes de tiempo, que parten con una función probabilística determinada. Para añadir un nuevo flujo se debe pulsar el botón "Añadir Ruta" y cumplimentar los distintos desplegables

con la información referente a la calle de inicio y final, y a la función de probabilidad que seguirá la distribución, así como los parámetros que requiera esta última (en vehículos por minuto). Además al pulsar el botón "Definir Tiempos" aparecerá una ventana similar a la que apareció para definir el tiempo de simulación, salvo que esta definirá la extensión temporal del flujo de vehículos generados. Al pulsar "Aceptar" se añadirá a la hoja principal la información facilitada, y se permitirá seguir añadiendo rutas hasta que se pulse "Cancelar". Con el botón "Borrar Ruta" de la primera hoja se permite seleccionar una ruta que aparezca en el desplegable, si la hubiera, y eliminarla.

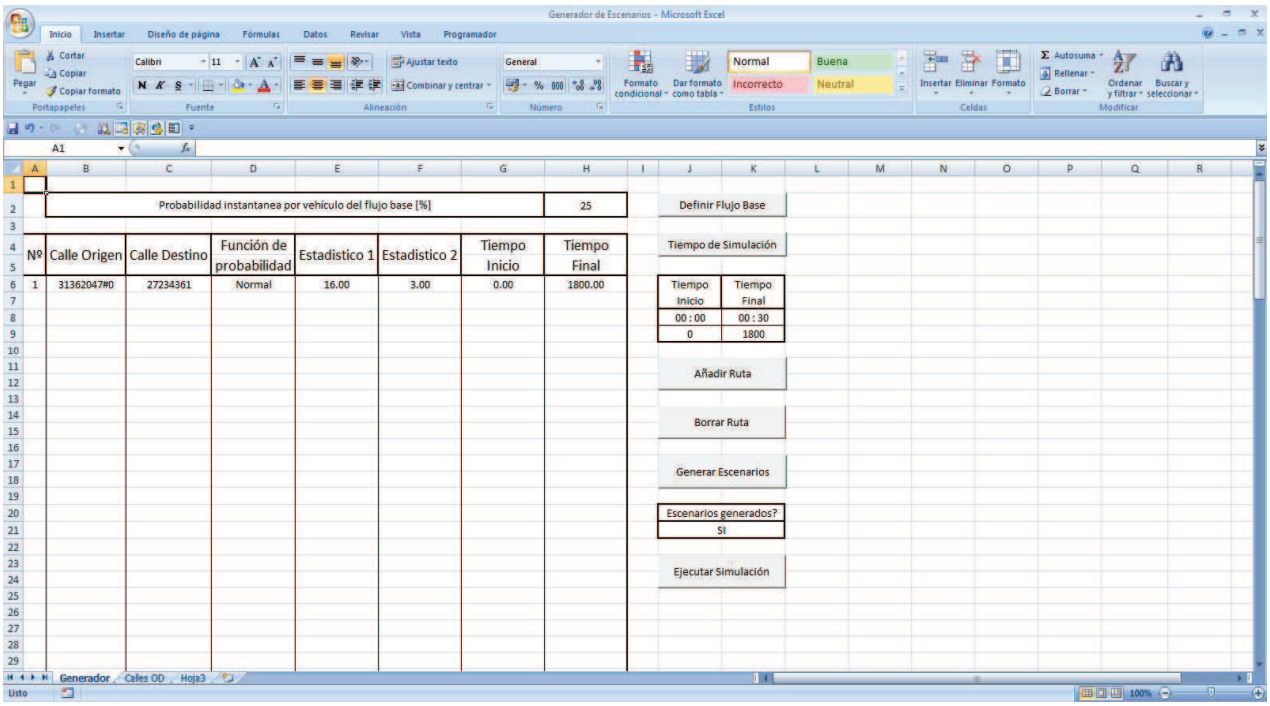


Ilustración 23. Detalle de la interfaz de usuario lista para llevar a cabo sobre ella una simulación

Por último, cuando se hayan añadido las rutas requeridas y se hayan fijado los distintos parámetros necesarios para la simulación se podrá pulsar el botón "Generar Escenarios" seguido del botón "Ejecutar Simulación". Este orden no puede alterarse ya que para ejecutar la simulación primero se deben crear ciertos ficheros a través del primer botón mencionado.

Para hacer todo lo anteriormente mencionado es necesario que se hayan incluido en la segunda hoja del libro de Excel ("Calles OD") todas las calles desde las cuales puede crearse una ruta y todas en la que podría morir. Bajo la columna con nombre "Calles Origen" se deben escribir el nombre exacto de las calles desde las que se desea que pueda salir un vehículo, y bajo la columna "Prob. Orig." un factor proporcional a la probabilidad que tendrán de salir desde la calle correspondiente. Bajo la columna con nombre "Calles Destino" se deben escribir el nombre exacto de las calles hasta las que se desea que pueda llegar un vehículo, y bajo la columna "Prob. Dest." un factor proporcional a la probabilidad que tendrán de llegar hasta la calle correspondiente. Hay que tener en cuenta que en el simulador los vehículos aparecen al inicio de la calle en cuestión, pero también desaparecen al inicio de la misma y no serán contabilizados para ningún cálculo de esa calle. Por otro lado, la probabilidad real de salir o entrar de cada calle será el factor asignado entre la suma de todos los factores de esa columna. Además bajo la columna "Funciones de Probabilidad" se escribirán las posibles funciones de distribución que pueden seguir los flujos así como el nombre de los parámetros o estadísticos que requiera esa función. En el caso de añadir nuevas funciones se deben de definir también las respectivas funciones de probabilidad en el programa "GeneradorRutas.py".[1]

Una vez hayamos pulsado el botón "Ejecutar Simulación" se abrirá el simulador, que nos mostrará el comportamiento del tráfico ante la estrategia utilizada. En caso de que esta simulación estuviese destinada a la toma de resultados, bastaría con extraer de la carpeta en la que se encuentra el control los archivos "Resultados.txt" y "Consumos.txt" que habrán sido creados a tal efecto.

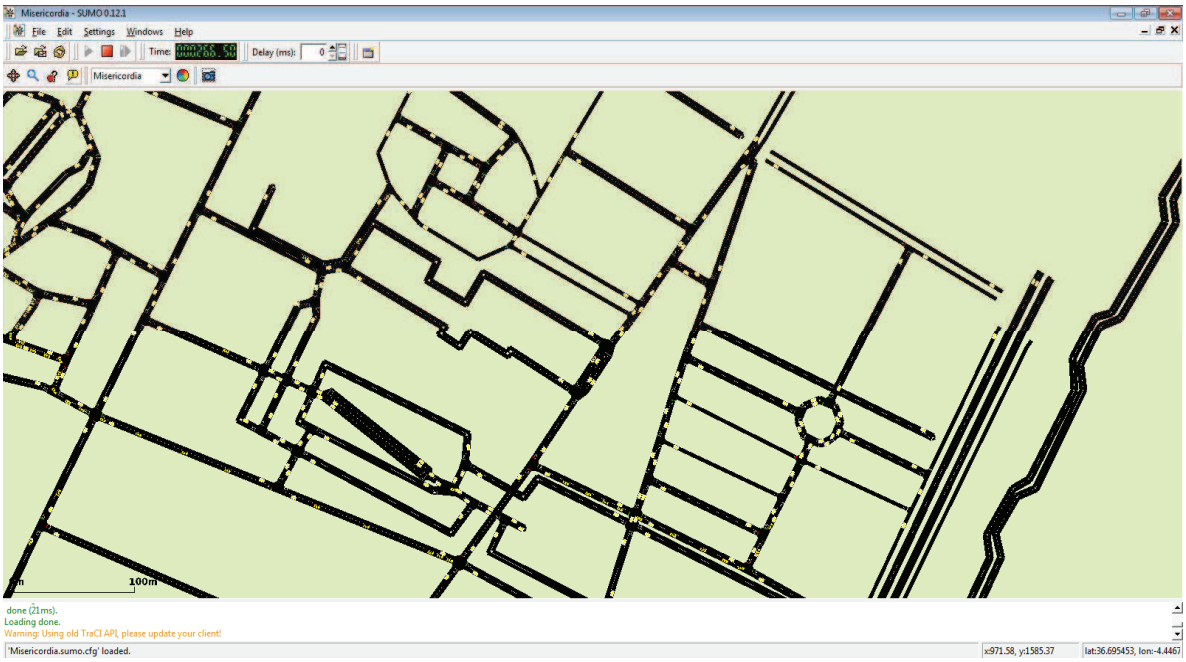


Ilustración 24. Detalle de una simulación en el Barrio de la Misericordia

UNIVERSIDAD PONTIFICIA COMILLAS
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)
INGENIERO INDUSTRIAL

Documento II

PRESUPUESTO

ÍNDICE

Capítulo 1 *Mediciones* ***5***

1 Equipos y software **5**

2 Mano de obra directa **5**

Capítulo 2 *Precios unitarios* ***7***

1 Equipos y software **7**

2 Mano de obra directa **7**

Capítulo 3 *Sumas parciales* ***9***

1 Equipos y software **9**

2 Mano de obra directa **9**

Capítulo 4 *Presupuesto general* ***11***

Capítulo 1 MEDICIONES

En este capítulo se recogen las unidades de los equipos utilizados así como las horas imputables al proyecto divididas en varias partidas diferentes.

1 Equipos y software

Elemento / Programa	Cantidad	Horas de Proyecto	Horas de uso al año
Ordenador	1	630	1.600
Simulador SUMO	1	400	1.000
Python	1	250	700
Microsoft Office 2007	1	80	400
IBM InfoSphere DataStage	1	70	1.000
IBM ILOG CPLEX	1	250	1.000

2 Mano de obra directa

Actividad	Horas
Documentación inicial	100
Desarrollo de la interfaz de usuario	10
Desarrollo del modelo matemático	100
Desarrollo del sistema de control	100
Desarrollo del método de implantación en nuevos escenarios	80
Pruebas y solución de problemas	170
Documentación del proyecto	100
Horas totales	660

Capítulo 2 PRECIOS UNITARIOS

En este capítulo se indican los precios de cada uno de los elementos necesitados para llevar a cabo este proyecto.

1 Equipos y software

Elemento / Programa	Precios (€/ud.)
Ordenador	2.200,00
Simulador SUMO	Software libre
Python	Software libre
Microsoft Office 2007	139,00
IBM InfoSphere DataStage	805,94
IBM ILOG CPLEX	9.207,54

2 Mano de obra directa

Actividad	Precio (€/h)
Documentación inicial	15,00
Desarrollo de la interfaz de usuario	25,00
Desarrollo del modelo matemático	40,00
Desarrollo del sistema de control	40,00
Desarrollo del método de implantación en nuevos escenarios	40,00
Pruebas y solución de problemas	50,00
Documentación del proyecto	15,00

Capítulo 3 SUMAS PARCIALES

En este capítulo se consignan los importes parciales de cada uno de los elementos que conforman las partidas. Estos importes han sido calculados a partir de las mediciones y los precios unitarios.

1 Equipos y software

Elemento / Programa	Uds	Horas de Proyecto	Horas por año	Precios (€/ud.)	Amortización Anual	Coste (€)
Ordenador	1	630	1.600	2.200,0	25 %	216,56
Microsoft Office 2007	1	80	400	139,00	25 %	6,95
IBM InfoSphere Data Stage	1	70	1.000	805,94	25 %	14,10
IBM ILOG CPLEX	1	250	1.000	9.207,5	25 %	575,47
					TOTAL	813,08

2 Mano de obra directa

Actividad	Horas	Precio (€/h)	Coste total (€)
Documentación inicial	100	15,00	1.500,00
Desarrollo de la interfaz de usuario	10	25,00	250,00
Desarrollo del modelo matemático	100	40,00	4.000,00
Desarrollo del sistema de control	100	40,00	4.000,00
Desarrollo del método de implantación en nuevos escenarios	80	40,00	3.200,00
Pruebas y solución de problemas	170	50,00	8.500,00
Documentación del proyecto	100	15,00	1.500,00
		TOTAL	22.950,00

Capítulo 4 PRESUPUESTO GENERAL

Sumando la contribución de todas las partidas anteriores se concluye que el coste del proyecto asciende a:

Concepto	Coste (€)
Equipos y software	813,08
Mano de obra directa	22.950,00
TOTAL	23.763,08